2020

Chemical reaction networks: Computability, complexity, and randomness

Xiang Huang
Iowa State University

Chemical reaction networks: Computability, complexity, and randomness

by

Xiang Huang

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Computer Science

Program of Study Committee:
Jack H. Lutz, Major Professor
Pavan Aduri
Soma Chaudhuri
David Fernadnez-Baca
James Lathrop
Timothy McNicholl

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2020

DEDICATION

To my wife and best friend Qiuyan Liao,

my newborn son Wenqiao Huang,

and Xiang Huang in a parallel universe,

where he works as a programmer in Beijing

and spends more time with his family and friends in China.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my advisor Dr. Jack H. Lutz, who guided on this journey.

I thank my former and current lab-mates in the Laboratory for Molecular Programming (LAMP): Adam Case, Donald Stull, Titus Klinge, Andrei Migunov, and Xiaoyuan Li. Their friendship made the journey full of joy. Adam introduced me effective dimension and mutual dimension when I first entered LAMP and knew nothing about these topics. Don and I published our first paper together. I will miss the days we spent in front of the chalk boards discussing Math. Titus talked to me about Chemical Reaction Networks (CRNs) even before I entered LAMP. I learned a lot of tricks on constructing CRNs from him. The workflow and software tools he introduced to me made me much more efficient. Andrei proofread a lot of my English writings, including my job applications, my postdoctoral fellowship proposal, and paper drafts. Xiaoyuan asked me a lot of questions, sometimes challenging ones, which often forced me to find better answers and better explanations. She actually brought the topic of CRN-computable numbers to my attention. It is a story worth telling.

It started with an exercise Jack assigned in his COM S 533 class that she took. I solved the exercise and found that it is not hard to compute algebraic numbers. Jim then challenged me to compute π. Titus and Jim first answered that through an experimental method and later I came up with a constructional solution. Of course, finally transforming everything into proofs took us a lot more time. It all started with an exercise problem and I ran into it by accident!

I thank Dr. Zhilin Wu in Institute of Software, Chinese Academy of Sciences, who encouraged me to go abroad and jump on this journey. I thank my former advisor Dr. Ting Zhang, who recruited me into his lab and supported my first year's study in Iowa State. I thank Dr. Simanta Mitra for nominating me for the Teaching Excellency Award. I thank all friends in the Computer Science Department, who made me feel at home.

I thank my colleagues Dr. David Voorhees and Dr. Aparna Das at Le Moyne College where I spent my last year as a visiting assistant professor. I learned a great deal about teaching from both of them. Dave also helped in cover my classes when I was away for job interviews and took a caring interest in my progress on teaching, research, and job hunting.

I thank Dr. Tim McNicholl, Dr. Jim Lathrop, and needless to say, Jack, for the enormous help in my academic job hunting in year 2019 and especially, in 2020, during this once-in-a-century pandemic. Many positions had been canceled due to COVID-19, including some that I applied and interviewed for. Luckily, I landed a tenure-track position at University of Illinois at Springfield before everything became much worse.

Last but not least, I'd like to thank my parents and siblings back in China. I am always indebted to them for their love and support.

ABSTRACT

In this dissertation we study the computational power of chemical reaction networks (CRNs), under both the deterministic and stochastic semantics of the model. We explore the class of real numbers that are computed in real time by deterministic CRNs. We develop the elements of the theory of algorithmic randomness in continuous-time Markov chains with the aim of applying this theory to stochastic CRNs, which are essentially special cases of CTMCs.

We first introduce the notion of computing a real number in real time. We show that every algebraic number is computable by chemical reaction networks in real time. We also show the real-time equivalent of CRNs and general purpose analog computers (GPACs), which are seemingly more powerful that CRNs. As a by-product of this fact, we give simple and natural constructions for some famous transcendental numbers.

Next we extend the above work to population protocols. We generalize the notion of numbers computed by large population protocols (LLPs) (Bournez, Fraigniaud, and Koegler, 2012). They proved that large population protocols can only compute exactly the algebraic numbers. However, their definition comes with an extra restriction: the systems must have finitely many fixed points. We relax the finitary restriction and show that we can now compute transcendental numbers.

Lastly, we discuss algorithmic randomness in continuous-time Markov chains. We first define the randomness of trajectories in terms of a new kind of martingale (algorithmic betting strategy). After that we prove equivalent characterizations in terms of constructive measure theory and Kolmogorov complexity. As a preliminary application we prove that, in any stochastic chemical reaction network, every random trajectory with bounded molecular counts has the non-Zeno property that infinitely many reactions do not occur in any finite interval of time.

CHAPTER 1. OVERVIEW

1.1 Real-Time CRN-computability

What is the computational power of a computational model? A simple and yet profound idea is to check which real numbers it can compute. In 1936, Turing introduced computable numbers [57] when he was considering the model that we now know as the Turing machine. Yamada [67] extended this notion by injecting a complexity notion into it and began to consider real-time computability, which later was proven to be equivalent to linear-time computability on a Turing machine [14]. Although we can see that rational numbers and some transcendental numbers, e.g. Liouville numbers, are fairly easy to compute by Turing machines in real time, a provoking question to ask is: Can we compute algebraic numbers, say, $\sqrt{2}$, in the same manner? Hartmanis and Stearns [24] conjectured no. To date, this is an open problem. We do not directly address the conjecture in this thesis, instead, we develop the notion of *real-time computability on analog models* and reveal some interesting facts about it.

What kind of *analog models* do we want to base our discussion on? In analog computing, there is a counterpart of Turing machines known as the general-purpose analog computer (GPAC), which was introduced by Shannon [51] in 1941. It can be simply characterized by polynomial ordinary differential equations (ODEs). We focus, however, on Chemical Reaction Networks (CRNs), a restricted form of GPACs, for their wide presence in molecular programming applications. Intriguingly, we will see that this choice does not introduce any difference in the class of numbers that can be computed in real time.

Another essential aspect is the notion of time, without which we can not have a correct notion of complexity or real-time computability. Note that the ODE systems associated with GPACs or CRNs do come with a natural parameter of time. But unfortunately, it does not suffice to use this sense of time directly since one can dilate it easily by applying the usual change of variables method.

A straightforward way to prevent that is to require all variables to be bounded. By doing this, one can only dilate the parameter of time for no more than a linear factor, which makes the notion of time robust.

We also need to prevent one from encoding too much information into the initial values of variables or the reaction rate constants in a CRN while computing a real number. Therefore we require all variables to start from zero at the beginning of time and similarly, all the reaction rates to be integral. We say a real number α is CRN-computable in real time, denoted as $\alpha \in \mathbb{R}_{RTCRN}$, if there is a CRN with integral rate constants and a designated species X such that, when all species concentrations are initialized to zero, X converges to α exponentially quickly.

By doing all of the above, whenever a CRN testifies that a real number α is CRN-computable in real time, the CRN serves as a finite description of the real number, or we can say it is an algorithm that computes α. That is, an algorithm which runs on an analog machine (a CRN).

As a main result in Chapter 2, we show that all algebraic numbers are in $\mathbb{R}_{RTCRN}$. This result might indicate a difference between analog models and discrete models, or maybe this difference is caused by the ways we define computing here: we do not produce or generate the target number bit by bit. One by-product in this chapter is the class of numbers we call *Lyapunov CRN-computable numbers*, which require the number being computed to be an *exponential stable fixed point* of the ODE associated with the CRN. We will discuss the requirement and its implications in Chapter 4 again.

In Chapter 3 we continue the investigation of $\mathbb{R}_{RTCRN}$. We show that the class forms a field, which gives us some structural information of the class and more importantly, it enables us to do modular design while constructing complicated CRNs. The main result in this chapter shows the real-time equivalence of CRNs and GPACs, which leads to concise and natural proofs that e, π, Euler's γ, and the Dottie number, among other interesting transcendental real numbers, are in $\mathbb{R}_{RTCRN}$.

1.2 Large Population Protocol Computable Numbers

Next, we shift our focus to Population Protocols (PPs) in Chapter 3. One can view PPs as restricted CRNs, in the sense that they conserve the total molecular count. More specifically, PPs require every reaction to have exactly two reactants and exactly two products. In [3, 30], the authors use Large Population Protocols (LPPs), a model that bridges stochastic (discrete, probabilistic) models and deterministic (continuous) models, to study the computational power of Population Protocols by, again, investigating the class of real numbers that LPPs compute. The underlying idea: when the population of a stochastic PP becomes very large, it behaves like a deterministic one. A more general theorem, the so-called Kurtz's theorem, is proven in [31]. This states that the phenomenon also applies to CRNs. So, roughly, to understand the long-term behavior of a stochastic PP when the population grows very large, one just need to turn to the corresponding deterministic one. In particular, when the PP contains fixed points, the authors use these fixed points to define a class of numbers that can be computed by LPPs. They showed that the class contains exactly the algebraic numbers. We will relax some of the restrictions in their definition and show that under the new definition one can now compute $\frac{\pi}{4}$, which makes the notion more interesting.

1.3 Algorithmic Randomness in Continuous-Time Markov Chain

Above, we mentioned Kurtz's theorem in LPPs, which says that the deterministic model is the infinite-volume limit of the stochastic model. The core of the proof of those theorems essentially involves measuring *sets* of trajectories that satisfy or violate certain properties. If *almost all* trajectories satisfy a property, in either the intuitive understanding, or more strictly, in the measure-theoretical understanding, one can roughly say that a random trajectory will likely satisfy the property. Note that this thinking jumps from talking about a *set* of objects to focusing on an *individual* random object, which sometimes be viewed as a more natural way to think about probability theory. But what exactly does "random" mean here? The theory of algorithmic randomness provides a robust answer. In order to talk about a single random trajectory in a

stochastic CRN, which is essentially a continuous-time Markov chain (CTMC), we need to first develop a good notion of randomness in CTMCs.

In Chapter 5 we develop the elements of such a theory. Our main contribution is a rigorous, useful notion of what it means for an *individual trajectory* of a CTMC to be *random*. CTMCs have discrete state spaces and operate in continuous time. This, together with the fact that trajectories may or may not halt, presents challenges not encountered in more conventional developments of algorithmic randomness.

After defining the randomness of trajectories in terms of a new kind of martingale, we prove equivalent characterizations in terms of constructive measure theory and Kolmogorov complexity. This shows the robustness of the definition.

As a preliminary application we prove that, in any stochastic chemical reaction network, *every* random trajectory with bounded molecular counts has the *non-Zeno property* that infinitely many reactions do not occur in any finite interval of time.

Materials

The main body of Chapters 2, 3, and 5 is taken from one journal article and two conference publications listed below.

- Xiang Huang, Titus H. Klinge, James I. Lathrop, Xiaoyuan Li and Jack H. Lutz: Real-Time Computability of Real Numbers by Chemical Reaction Networks. *Volume 18, Issue 1, pp 63-73, Natural Computing (2019).* ***(invited paper).***

- Xiang Huang, Jack H. Lutz, and Andrei N. Migunov. Algorithmic Randomness in Continuous-Time Markov Chains, 2019. In Proceedings of the 57th Annual Allerton Conference on Communication, Control, and Computing.

- Xiang Huang, Titus H. Klinge, James I. Lathrop. Real-Time Equivalence of Chemical Reaction Networks and Analog Computers. In: Thachuk C., Liu Y. (eds) DNA Computing

and Molecular Programming. DNA 2019. Lecture Notes in Computer Science, vol 11648. Springer, Cham.

The reader should note that these papers contain collective efforts of all my coauthors, even though I consider myself to have played an active role. I also made some modifications and useful comments for better incorporation of these papers into this thesis.

In Section 3.4, the computation of Euler's γ and the Dottie number are unpublished results. So is Chapter 4. I am solely responsible for any mistakes that might occur in these sections or chapters.

CHAPTER 2. REAL-TIME CRN COMPUTABILITY

2.1 Introduction

Chemical reaction networks, originally conceived as descriptive mathematical models of molecular interactions in well-mixed solutions, are also widely used as prescriptive mathematical models for engineering molecular processes. In the present century this prescriptive use of chemical reaction networks has been automated by software compilers that translate chemical reaction networks into complete specifications of DNA strand displacement systems that simulate them [55, 8]. Chemical reaction networks have thus become the programming language of choice for many molecular programming applications.

There are several alternative semantics (operational meanings, also called kinetics) for chemical reaction networks. The two oldest and most widely used of these are *deterministic mass-action semantics* and *stochastic mass-action semantics*. This chapter and Chapter 3 concern the former of these, so for the rest of this chapter, a *chemical reaction network* (briefly, a *CRN* or a *deterministic CRN*) is a chemical reaction network with deterministic mass-action semantics. This model is precisely specified in section 2 below. For this introduction, it suffices to say that such a CRN is an ordered pair $N = (S, R)$, where S is a finite set of *species* (abstract molecule types), and R is a finite set of *reactions*, each of which has some form like:

$$X + Z \xrightarrow{k} 2Y + Z,$$

where $X, Y, Z \in S$ are species and $k \in (0, \infty)$ is a *rate constant*. A *state* $\mathbf{x}$ of N specifies the real-valued *concentration* $\mathbf{x}(Y) \in [0, \infty)$ of each species $Y \in S$. Given an initial state $\mathbf{x}(0)$ at time $t = 0$, deterministic mass action semantics specify the (continuous) evolution of the state $\mathbf{x}(t)$ over time.

Even prior to the implementation of chemical reaction networks as a programming language it was clear that they constitute a model of computation. In the case of deterministic CRNs, Stansifer

had reportedly proven [9, 53] that this model is Turing universal, i.e., that every algorithm can be simulated by a deterministic CRN, but no proof was published. (Note: The title of [41] seems to make this assertion, but the paper only exhibits a way to use deterministic CRNs to simulate finite Boolean circuits.) Fages, Le Guludec, Bournez, and Pouly [12] have now proven this universality theorem.

Deterministic chemical reaction networks are an analog model of computation, both in the intuitive sense that their states are vectors of real-valued concentrations that vary continuously over real-valued times and in the technical sense that they are a special case of Shannon's *general purpose analog computer* (*GPAC*) [51], as explained in section 5 below.

This chapter studies the ability of deterministic CRNs to rapidly compute real numbers in the following analog sense. We say that a deterministic CRN *computes* a real number α *in real time* if it has a designated species X such that the following three things hold. (See section 3 for more details.) First, the CRN's reaction rate constants are positive integers, and it is initialized with all concentrations set to zero at time $t = 0$. This implies that the CRN is, like any reasonable model of computation, finitely specifiable. It also implies that only countably many real numbers are real time CRN-computable. Second, there is some fixed bound on all the CRN's concentrations. Under deterministic mass-action semantics, this implies that all the reaction rates of the CRN are bounded, whence time is a meaningful resource. Third, the concentration $x(t)$ of the designated species $X(t)$ is within 2^{-t} of $|\alpha|$ – i.e., within t bits of accuracy of $|\alpha|$ – at all times $t \geq 1$. We say that the real number α is *real time computable by chemical reaction networks* (briefly, *real time CRN-computable*) if there is a CRN that computes α in this sense. Elementary properties of real-time CRN computability are developed in section 3.

Our main theorem in this chapter says that every algebraic number (i.e., every real solution of a polynomial with integer coefficients) is real time CRN-computable. This result is proven in sections 4 and 5.

Section 6 contains two discussions. First, we compare real-time CRN computability with computability in the closely related *large population protocol (LPP)* model of Bournez, Fraigniaud,

and Koegler [3, 30]. Second, our main theorem is a counterpoint – but not a disproof – of the 60-year-old, open Hartmanis-Stearns conjecture that no algebraic irrational is real time computable by a Turing machine [24]. We discuss this contrast in some detail and pose two questions whose answers would shed further light on the computational complexities of algebraic irrationals.

2.2 Chemical Reaction Networks

A *species* is an abstract type of molecule. Capital Roman characters such as X, Y, and Z are commonly used to distinguish different species, but we also use decorations such as X_0, $\widehat{Y}$, and $\overline{Z}$ to distinguish them.

A *reaction* over a finite set S of species is a tuple $\rho = (\mathbf{r}, \mathbf{p}, k) \in \mathbb{N}^S \times \mathbb{N}^S \times (0, \infty)$ and its components are called the *reactant vector*, the *product vector*, and the *rate constant*, respectively. (Here $\mathbb{N}^S$ denotes the set of all functions mapping S into $\mathbb{N}$.) To avoid excessive use of subscripts, for a reaction ρ we use $\mathbf{r}(\rho)$, $\mathbf{p}(\rho)$, and $k(\rho)$ to access the individual components of ρ. A species $Y \in S$ is called a *reactant* if $\mathbf{r}(Y) > 0$, called a *product* if $\mathbf{p}(Y) > 0$, and called a *catalyst* if $\mathbf{r}(Y) = \mathbf{p}(Y) > 0$. The *net effect* of reaction $\rho = (\mathbf{r}, \mathbf{p}, k)$ is the vector $\Delta\rho \in \mathbb{N}^S$ defined by

$$\Delta\rho(Y) = \mathbf{p}(Y) - \mathbf{r}(Y)$$

for each $Y \in S$.

A *chemical reaction network* (*CRN*) is an ordered pair $N = (S, R)$ where S is a finite set of species and R is a finite set of reactions over S. Although this completes the definition of the *syntax* of a CRN, we have yet to define the *semantics* used in this chapter.

Under *deterministic mass action semantics*, the *state* of a CRN $N = (S, R)$ at time t is a real-valued vector $\mathbf{x}(t) \in [0, \infty)^S$, and for $Y \in S$, we call $\mathbf{x}(t)(Y)$ the *concentration* of Y in $\mathbf{x}(t)$. We also write $y(t) = \mathbf{x}(t)(Y)$ to denote the concentration of species Y at time t.

The *rate* of a reaction ρ at time t is defined as

$$\text{rate}_\rho(t) = k(\rho) \cdot \prod_{Y \in S} y(t)^{\mathbf{r}(\rho)(Y)}. \tag{2.2.1}$$

This conforms to the so-called law of mass action which states that the rate of a reaction is proportional to the concentration of its reactants.

The *total rate of change* of a species $Y \in S$ depends on the rates of all reactions in the CRN and the magnitude of their net effect on Y. Therefore the concentration $y(t)$ conforms to the ordinary differential equation (ODE)

$$\frac{dy}{dt} = \sum_{\rho \in R} \Delta\rho(Y) \cdot \text{rate}_\rho(t). \tag{2.2.2}$$

If we let $\mathcal{E}_Y$ be the ODE above for each $Y \in S$, then the *mass action system* of the CRN is the coupled system

$$(\mathcal{E}_Y \mid Y \in S). \tag{2.2.3}$$

Given an initial state $\mathbf{x}_0 \in [0, \infty)^S$, the behavior of the CRN is defined as the solution to the initial value problem (IVP) of the mass action system (2.2.3) along with the initial condition

$$y(0) = \mathbf{x}_0(Y)$$

for each $Y \in S$.

2.3 Real-Time CRN Computability

We say that a real number α is *real time computable by chemical reaction networks* (briefly, *real time CRN-computable*), and we write $\alpha \in \mathbb{R}_{RTCRN}$, if there exist a chemical reaction network $N = (S, R)$ and a species $X \in S$ with the following three properties:

1 (integrality). The CRN N is *integral* in the sense that:

$$k(\rho) \in \mathbb{Z}^+ \tag{2.3.1}$$

for all $\rho \in R$.

2 (boundedness). There is a constant $\beta > 0$ such that, if N is initialized with $y(0) = 0$ for all $Y \in S$, then, for all $Y \in S$ and $t \in [0, \infty)$,

$$y(t) \leq \beta. \tag{2.3.2}$$

3 (real-time convergence). If N is initialized with $y(0) = 0$ for all $Y \in S$, then for all $t \in [1, \infty)$,

$$|x(t) - |\alpha|| \leq 2^{-t}. \tag{2.3.3}$$

The integrality condition (2.3.1) prevents the CRN N from "cheating" by having information about α explicitly encoded into its rate constants. To see that this is necessary to avoid nontriviality, note that, for any $\alpha \in (0, \infty)$, if the simple CRN:

$$\emptyset \xrightarrow{\alpha} X,$$
$$X \xrightarrow{1} \emptyset$$

is initialized with $x(0) = 0$, then

$$x(t) = \alpha(1 - e^{-t})$$

for all $t \in [0, \infty)$.

The boundedness condition (2.3.2) imposes a "speed limit" on the CRN N. This prevents N from acting as a "Zeno machine" (machine that does infinite work in finite time) in the sense of Weyl [66]. More precisely, condition (2.3.2) ensures that the reaction rates (2.2.1) of N are all bounded. This implies that the arc length of the curve traced by the state $\mathbf{x}(s)$ of N for $0 \leq s \leq t$ is $\theta(t)$, i.e., bounded above and below by positive constant multiples of t. Pouly [45, 4] has convincingly argued (in a more general setting) that this arc length, which we call the *reaction clock time*, is the correct measure of the time that a CRN spends computing during the interval $[0, t]$. Viewed in this light, condition (2.3.2) ensures that t is, up to constant multiples, an accurate measure of the reaction clock time of N during the interval $[0, t]$.

The real-time convergence condition (2.3.3) requires the CRN N to compute $|\alpha|$ to within t bits of accuracy by each time $t \geq 1$. Note that this is an *analog* approximation of $|\alpha|$. The CRN N is not required to explicitly produce symbols in any sort of digital representation of $|\alpha|$.

For the rest of this chapter, unless otherwise noted, all CRNs $N = (S, R)$ are assumed to be initialized with $y(0) = 0$ for all $Y \in S$.

To save space in our first lemma, we define the predicate

$$\begin{aligned}\Phi_{\tau,\gamma}(\alpha) \equiv\ &\text{there exist a CRN } N = (S, R) \text{ and a species } X \in S \\ &\text{satisfying (2.3.1) and (2.3.2) such that, for all } t \in [\tau, \infty), \\ &\big|x(t) - |\alpha|\big| \leq e^{-\gamma t}\end{aligned}$$

for each $\tau, \gamma \in (0, \infty)$ and $\alpha \in \mathbb{R}$. Note that $\Phi_{1,\ln 2}(\alpha)$ is the assertion that $\alpha \in \mathbb{R}_{RTCRN}$. The following convenient lemma says that the definition of $\mathbb{R}_{RTCRN}$ is robust with respect to linear changes in condition (2.3.2).

Lemma 2.3.1. *For each $\alpha \in \mathbb{R}$ the following conditions are equivalent.*

1. $\alpha \in \mathbb{R}_{RTCRN}$.
2. *There exists $\tau, \gamma \in (0, \infty)$ such that $\Phi_{\tau,\gamma}(\alpha)$ holds.*
3. *For every $\tau, \gamma \in (0, \infty)$, $\Phi_{\tau,\gamma}(\alpha)$ holds.*

Proof. Let $\alpha \in \mathbb{R}$. It is clear that (3) $\Rightarrow$ (1) $\Rightarrow$ (2), so it suffices to prove that (2) $\Rightarrow$ (3). For this, let N, X, τ, and γ testify that (2) holds, i.e., let N and X testify that $\Phi_{\tau,\gamma}(\alpha)$ holds. To prove (3), let $\widehat{\tau}, \widehat{\gamma} \in (0, \infty)$. It suffices to show that $\Phi_{\widehat{\tau},\widehat{\gamma}}(\alpha)$ holds. Let

$$a = \max\left\{\left\lceil\frac{\tau}{\widehat{\tau}}\right\rceil, \left\lceil\frac{\widehat{\gamma}}{\gamma}\right\rceil\right\},$$

and let $\widehat{N} = (S, \widehat{R})$, where

$$\widehat{R} = \{(\mathbf{r}, \mathbf{p}, ak) \mid (\mathbf{r}, \mathbf{p}, k) \in R\}.$$

That is, $\widehat{N}$ is exactly like N, except that each rate constant of N has been multiplied by the positive integer a. Then $\widehat{N}$ is an integral CRN that is a "sped up version" of N in the sense that, for all $y \in S$ and $t \in [0, \infty)$,

$$y_{\widehat{N}}(t) = y_N(at), \tag{2.3.4}$$

where y_N and $y_{\widehat{N}}$ are the values of y in N and $\widehat{N}$, respectively. This immediately implies that $\widehat{N}$ satisfies (2.3.2). Now let $t \in [\widehat{\tau}, \infty)$. Then $at \in [\tau, \infty)$, so our assumption $\Phi_{\tau,\gamma}(\alpha)$ tells us that

$$\begin{aligned} |x_{\widehat{N}}(t) - |\alpha|| &= |x_N(at) - |\alpha|| \\ &\leq e^{-\gamma at} \\ &\leq e^{-\widehat{\gamma} t}, \end{aligned}$$

affirming $\Phi_{\widehat{\tau},\widehat{\gamma}}(\alpha)$. □

The following lemma is a warm-up for our examination of $\mathbb{R}_{RTCRN}$

Lemma 2.3.2. $\mathbb{Q} \subsetneqq \mathbb{R}_{RTCRN}$

Proof. If $\alpha = 0$, then the CRN $N = (\{X\}, \emptyset)$ testifies that $\alpha \in \mathbb{R}_{RTCRN}$. If $\alpha \in \mathbb{Q} \setminus \{0\}$, then we can write $|\alpha| = \frac{a}{b}$, where $a, b \in \mathbb{Z}^+$. Then the integral CRN

$$\begin{aligned} \emptyset &\xrightarrow{a} X \\ X &\xrightarrow{b} \emptyset \end{aligned}$$

satisfies

$$x(t) = \frac{a}{b}(1 - e^{-bt}),$$

so $\alpha \in \mathbb{R}_{RTCRN}$ by Lemma 2.3.1. This shows that $\mathbb{Q} \subseteq \mathbb{R}_{RTCRN}$.

To see that $\mathbb{Q} \neq \mathbb{R}_{RTCRN}$, it suffices to show that $\frac{1}{\sqrt{2}} \in \mathbb{R}_{RTCRN}$. Since the integral CRN

$$\begin{aligned} \emptyset &\xrightarrow{1} X \\ 2X &\xrightarrow{2} X \end{aligned}$$

satisfies

$$x(t) = \frac{1}{\sqrt{2}} \left(\frac{1 - e^{-2\sqrt{2}t}}{1 + e^{-2\sqrt{2}t}} \right),$$

we have that

$$\begin{aligned} |x(t) - \frac{1}{\sqrt{2}}| &= \frac{1}{\sqrt{2}} \left(\frac{2e^{-2\sqrt{2}t}}{1 + e^{-2\sqrt{2}t}} \right) \\ &\leq \sqrt{2} e^{-2\sqrt{2}t}, \end{aligned}$$

so $\frac{1}{\sqrt{2}} \in \mathbb{R}_{RTCRN}$ by Lemma 2.3.1. □

Computable real numbers were introduced by Turing [57, 58] and have been extensively investigated [29, 64].

A real number α is *computable*, and we write $\alpha \in \mathbb{R}_{comp}$, if there is a computable function $\widehat{\alpha} : \mathbb{N} \to \mathbb{Q}$ such that, for all $r \in \mathbb{N}$

$$|\widehat{\alpha}(r) - \alpha| \leq 2^{-r}.$$

Lemma 2.3.3. $\mathbb{R}_{RTCRN} \subsetneqq \mathbb{R}_{comp}$

Proof. Let $\alpha \in \mathbb{R}_{RTCRN}$, and let $N = (S, R)$ and $X \in S$ testify to this fact. Let $Y_1, \ldots, Y_n$ be the distinct species in S. Then the ODEs (2.2.2) can be written in the form

$$\begin{aligned} y_1' &= f_1(y_1, \ldots, y_n), \\ &\vdots \\ y_n' &= f_n(y_1, \ldots, y_n), \end{aligned} \tag{2.3.5}$$

where $f_1, \ldots, f_n$ are polynomials with integer coefficients. By the boundedness condition (2.3.2) and Theorem 16 of [19], the solution $\mathbf{y} : [0, \infty) \to [0, \infty)^n$ of (2.3.5) is polynomial time computable. It follows by the real-time convergence condition (2.3.3) that α is computable in polynomial time in the sense of Ko [29]. Hence, $\alpha \in \mathbb{R}_{comp}$.

It is well known [29] that not every computable real is computable in polynomial time, so the preceding paragraph proves the lemma. □

2.4 Lyapunov CRN Computability

This section defines a subclass of $\mathbb{R}_{RTCRN}$, namely, the class $\mathbb{R}_{LCRN}$ of all Lyapunov CRN-computable real numbers. The main theorem of this section is the fact that $\mathbb{R}_{LCRN}$ is a field.

Our definition of $\mathbb{R}_{LCRN}$ uses the stability theory of ordinary differential equations. We review the elements of this theory that we need here, referring the reader to standard textbooks (e.g., [25, 56]) for more thorough treatments.

We first note that the ordinary differential equations (2.2.2) of a CRN $N = (S, R)$ are *autonomous*, meaning that they only depend on the time t via the species concentrations $y(t)$. Hence, if we let $Y_1, \ldots, Y_n$ be the distinct species in S, then the ODEs (2.2.2) can be written as

$$\begin{aligned} y_1' &= f_1(y_1, \ldots, y_n), \\ &\vdots \\ y_n' &= f_n(y_1, \ldots, y_n), \end{aligned} \tag{2.4.1}$$

where $f_1, \ldots, f_n : \mathbb{R}^n \to \mathbb{R}$ are polynomials. If we let $\mathbf{f}_N : \mathbb{R}^n \to \mathbb{R}^n$ be the function whose components are $f_1, \ldots, f_n$, then (2.4.1) can be written in the vector form

$$\mathbf{x}' = \mathbf{f}_N(\mathbf{x}). \tag{2.4.2}$$

A *fixed point* of the CRN N is a state $\mathbf{z} \in [0, \infty)^S$ such that $\mathbf{f}_N(\mathbf{z}) = 0$. A state $\mathbf{z}$ of N is *exponentially stable* if there exist $\alpha, \delta, C \in (0, \infty)$ such that, for all $\mathbf{x}_0 \in [0, \infty)^S$ with $|\mathbf{x}_0 - \mathbf{z}| \leq \delta$, if N is initialized with $\mathbf{x}(0) = \mathbf{x}_0$, then, for all $t \in [0, \infty)$,

$$|\mathbf{x}(t) - \mathbf{z}| \leq Ce^{-\alpha t}|\mathbf{x}(0) - \mathbf{z}|. \tag{2.4.3}$$

It is easy to see that an exponentially stable state of N must be a fixed point of N.

In this chapter, we define a real number α to be *Lyapunov computable by chemical reaction networks* (briefly, *Lyapunov CRN-computable*), and we write $\alpha \in \mathbb{R}_{LCRN}$, if there exist a CRN $N = (S, R)$, a species $X \in S$, and a state $\mathbf{z} \in [0, \infty)^S$ with $\mathbf{z}(X) = |\alpha|$ such that the following conditions hold.

1 (integrality). The CRN N is integral as in (2.3.1).

2 (boundedness). Concentrations are bounded as in (2.3.2).

3 (exponential stability). $\mathbf{z}$ is an exponentially stable state of N.

4 (convergence). If $\mathbf{x}(t) \in [0, \infty)^S$ is the state of N at time t, then

$$\mathbf{x}(0) = \mathbf{0} \Longrightarrow \lim_{t \to \infty} \mathbf{x}(t) = \mathbf{z}.$$

(Here $\mathbf{0}$ is the state of N defined by $\mathbf{0}(Y) = 0$ for all $Y \in S$.)

A well known matrix characterization of exponential stability is useful for investigating the set R_{LCRN}.

The *Jacobian matrix* of the CRN N is the Jacobian matrix of $\mathbf{f}_N$, i.e., the $n \times n$ matrix

$$J_N = \begin{pmatrix} \frac{\partial f_1}{\partial y_1} & \cdots & \frac{\partial f_1}{\partial y_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial y_1} & \cdots & \frac{\partial f_n}{\partial y_n} \end{pmatrix}.$$

More precisely, the *Jacobian matrix* of N *in a state* $\mathbf{x} \in [0,\infty)^S$ is the matrix $J_N(\mathbf{x})$ in which each of the partial derivatives in J_N is evaluated at the point $\mathbf{x}$. The *eigenvalues* of the CRN N *in a state* $\mathbf{x} \in [0,\infty)^S$ are the eigenvalues of the matrix $J_N(\mathbf{x})$, i.e., the numbers $\lambda \in \mathbb{C}$ for which there exists $\mathbf{y} \neq \mathbf{0} \in \mathbb{R}^n$ such that $J_N(\mathbf{x})(\mathbf{y}) = \lambda \mathbf{y}$.

Lyapunov's *exponential stability theorem*, specialized to CRNs, says the following.

Theorem 2.4.1. *[40, 25] A fixed point* $\mathbf{z}$ *of a CRN* N *is exponentially stable if and only if all the eigenvalues of* N *in state* $\mathbf{z}$ *have negative real parts.*

We note the following easy inclusions.

Lemma 2.4.2. $\mathbb{Q} \subseteq \mathbb{R}_{LCRN} \subseteq \mathbb{R}_{RTCRN}$.

Proof. To see that $\mathbb{Q} \subseteq \mathbb{R}_{LCRN}$, it suffices to show that $[0,\infty) \cap \mathbb{Q} \subseteq \mathbb{R}_{LCRN}$. For this, let $\alpha \in [0,\infty) \cap \mathbb{Q}$. If $\alpha = 0$, then a one-species CRN with no reactions trivially affirms that $\alpha \in \mathbb{R}_{LCRN}$, so assume that $\alpha > 0$. Then there exist $p, q \in \mathbb{Z}^+$ such that $\alpha = \frac{p}{q}$. The CRN

$$\emptyset \xrightarrow{p} X$$

$$X \xrightarrow{q} \emptyset$$

computes α with species X. The ODE for this CRN is

$$x' = p - qx$$

and the solution for the initial value $x(0) = 0$ is $x(t) = -\frac{p}{q}e^{-qt} + \frac{p}{q}$. We thus have $\lim_{t\to\infty} x(t) = \alpha$. Moreover, since we have an ODE system with only one variable, the eigenvalue at the fixed point is simply the derivative of $p - qx$, i.e., $-q$. Hence $\alpha \in \mathbb{R}_{LCRN}$.

The inclusion $\mathbb{R}_{LCRN} \subseteq \mathbb{R}_{RTCRN}$ is immediate from the definitions of these classes and Lemma 2.3.1. □

The rest of this section proves that $\mathbb{R}_{LCRN}$ has the closure properties of a field.

Lemma 2.4.3. *For all $\alpha, \beta \in [0, \infty) \cap \mathbb{R}_{LCRN}$, we have $\alpha + \beta \in \mathbb{R}_{LCRN}$.*

Proof. Let $\alpha, \beta \in [0, \infty) \cap \mathbb{R}_{LCRN}$, and let $N_1 = (S_1, R_1)$ and $N_2 = (S_2, R_2)$ be CRNs that testify to this by computing α and β with species X and Y, respectively. We also let $\mathbf{z}_1 \in [0, \infty)^{S_1}$ and $\mathbf{z}_2 \in [0, \infty)^{S_2}$ be the equilibrium points that N_1 and N_2 use to compute α and β, i.e., $\mathbf{z}_1(X) = \alpha$ and $\mathbf{z}_2(Y) = \beta$.

Let $N = (S, R)$ be the CRN defined by $S = S_1 \cup S_2 \cup S_3$ and $R = R_1 \cup R_2 \cup R_3$ where $S_3 = \{U\}$ and R_3 consists of the three reactions

$$X \xrightarrow{1} X + U$$
$$Y \xrightarrow{1} Y + U$$
$$U \xrightarrow{1} \emptyset.$$

Note that the ODE for U is

$$\frac{du}{dt} = x + y - u,$$

and the solution for $u(t)$ with all species initialized to zero is

$$u(t) = ce^{-t} + e^{-t} \int_1^t e^s (x(s) + y(s)) ds,$$

for some constant c. Hence,

$$\lim_{t \to \infty} u(t) = 0 + \lim_{t \to \infty} \frac{\int_1^t e^s (x(s) + y(s)) ds}{e^t}.$$

By L'Hôpital's rule, we have

$$\lim_{t \to \infty} u(t) = \lim_{t \to \infty} [x(t) + y(t)] = \alpha + \beta.$$

It remains to be shown that the equilibrium point $(\mathbf{z}_1, \mathbf{z}_2, \alpha + \beta)$ is exponentially stable. First, we fix an order S_1, S_2, S_3 of the species in the Jacobian matrix J_N. We use $J_{i,j}$ to denote submatrix

of J_N that contains the partial derivatives of each species $A \in S_i$ with respect to each species $B \in S_j$ for $i, j \in \{1, 2, 3\}$. Then J_N can be written as follows.

$$J_N = \begin{pmatrix} J_{N_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & J_{N_2} & \mathbf{0} \\ J_{3,1} & J_{3,2} & J_{3,3} \end{pmatrix}$$

Since N_1 and N_2 have disjoint species, it is clear that $J_{1,2} = \mathbf{0}$ and $J_{2,1} = \mathbf{0}$. Furthermore, N_1 and N_2 are unaffected by the species Y, so $J_{1,3} = \mathbf{0}$ and $J_{2,3} = \mathbf{0}$. We also note that $J_{3,3}$ contains one element $\frac{\partial}{\partial u}(x + y - u) = -1$.

Since J_N is a lower triangular block matrix,

$$|J - \lambda I| = |J_{N_1} - \lambda I_1| \ \cdot \ |J_{N_2} - \lambda I_2| \ \cdot \ |-1 - \lambda| \tag{2.4.4}$$

We can now conclude that the eigenvalues of J_N are

1) the eigenvalues of J_{N_1},

2) the eigenvalues of J_{N_2}, and

3) the eigenvalue of $J_{3,3}$, which is -1.

Since α, β are both in $\mathbb{R}_{LCRN}$, the real part of the eigenvalues of J_{N_1} and J_{N_2} are all negative. Thus all the eigenvalues of J_N have negative real parts. Therefore, $\alpha + \beta \in \mathbb{R}_{LCRN}$. □

Lemma 2.4.4. *For each $0 < \alpha \in \mathbb{R}_{LCRN}$, we have $\frac{1}{\alpha} \in \mathbb{R}_{LCRN}$.*

Proof. Let $\alpha \in (0, \infty) \cap \mathbb{R}_{LCRN}$, and let $N_1 = (S_1, R_1)$ be a CRN that testifies to this by computing α with species X. We also let $\mathbf{z}_1 \in [0, \infty)^{S_1}$ be the equilibrium point that N_1 uses to compute α, i.e., $\mathbf{z}_1(X) = \alpha$.

Let $N = (S, R)$ be the CRN with $S = S_1 \cup S_2$ and $R = R_1 \cup R_2$ where $S_1 = \{Y\}$ and R_2 consists of the reactions

$$\emptyset \xrightarrow{1} Y$$

$$X + Y \xrightarrow{1} X$$

so that

$$\frac{dy}{dt} = 1 - xy.$$

It is easy to see that the state $\mathbf{z} = \left(\mathbf{z}_1, \frac{1}{\alpha}\right)$ is the only reachable equilibrium point of N from the initial state $\mathbf{0}$. We also note that the Jacobian matrix J_N can be written as

$$J_N = \begin{pmatrix} J_{N_1} & \mathbf{0} \\ -y & -x \end{pmatrix}.$$

We can see that $J_N(\mathbf{z})$ has eigenvalues with negative real parts, since the eigenvalues of J_{N_1} have negative real parts and $-\alpha < 0$. This implies that $\mathbf{z}$ is exponentially stable, and therefore $\frac{1}{\alpha} \in \mathbb{R}_{LCRN}$. □

Lemma 2.4.5. *For each $\alpha, \beta \in \mathbb{R}_{LCRN}$ with $\alpha \geq \beta \geq 0$, we have $\alpha - \beta \in \mathbb{R}_{LCRN}$.*

Proof. Let α and β be as given. If $\beta = 0$, then $\alpha - \beta = \alpha \in \mathbb{R}_{LCRN}$ by hypothesis. If $\beta = \alpha$, then $\alpha - \beta \in \mathbb{R}_{LCRN}$ by Lemma 2.4.2. Assume, then, that $\alpha > \beta > 0$, and let $N_1 = (S_1, R_1)$ and $N_2 = (S_2, R_2)$ testify that $\alpha, \beta \in \mathbb{R}_{LCRN}$ using species $X_1 \in S_1$ and $X_2 \in S_2$, respectively. We also let $\mathbf{z}_1$ and $\mathbf{z}_2$ be the fixed points in N_1 and N_2 that compute α and β, respectively.

Let $N = (S, R)$ be the CRN defined by $S = S_1 \cup S_2 \cup \{Y\}$ and $R = R_1 \cup R_2 \cup R_3$ where R_3 consists of the three reactions

$$\emptyset \xrightarrow{1} Y$$

$$X_1 + Y \xrightarrow{1} X_1$$

$$X_2 + Y \xrightarrow{1} X_2 + 2Y.$$

Note that the additional reactions do not affect the species in S_1 and S_2, and yield the following ODE for Y.

$$\frac{dy}{dt} = 1 - (x_1 - x_2)y.$$

If N is initialized to the $\mathbf{0}$ state, then by the construction of N, $\lim_{t\to\infty}(x_1(t) - x_2(t)) = \alpha - \beta$. It is then easy to show that $\mathbf{z} = (\mathbf{z}_1, \mathbf{z}_2, \frac{1}{\alpha-\beta})$ is a fixed point of N and that $\lim_{t\to\infty} y(t) = \frac{1}{\alpha-\beta}$. The

Jacobian matrix J_N also has the form,

$$J_N = \begin{pmatrix} J_{N_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & J_{N_2} & \mathbf{0} \\ J_{3,1} & J_{3,2} & -(x_1 - x_2) \end{pmatrix}$$

and therefore $J_N(\mathbf{z})$ has all negative eigenvalues since J_{N_1} and J_{N_2} have negative eigenvalues by the hypothesis and the eigenvalue $-(\alpha - \beta)$ is negative. Thus, $\frac{1}{\alpha-\beta} \in \mathbb{R}_{LCRN}$.

Finally, by Lemma 2.4.4 we can conclude that $\alpha - \beta \in \mathbb{R}_{LCRN}$. □

Corollary 2.4.6. *$\mathbb{R}_{LCRN}$ is an additive subgroup of $\mathbb{R}$.*

Proof. Lemma 2.4.2 tells us that $0 \in \mathbb{R}_{LCRN}$, and the definition of $\mathbb{R}_{LCRN}$ implies that it is closed under additive inverses. To see that $\mathbb{R}_{LCRN}$ is closed under addition, let $\alpha, \beta \in \mathbb{R}_{LCRN}$. Then either

$$|\alpha + \beta| = |\alpha| + |\beta|,$$

in which case $\alpha + \beta \in \mathbb{R}_{LCRN}$ by Lemma 2.4.3, or

$$|\alpha + \beta| = \max\{|\alpha|, |\beta|\} - \min\{|\alpha|, |\beta|\},$$

in which case $\alpha + \beta \in \mathbb{R}_{LCRN}$ by Lemma 2.4.5. □

Lemma 2.4.7. *For each $\alpha, \beta \in [0, \infty) \cap \mathbb{R}_{LCRN}$, we have $\alpha\beta \in \mathbb{R}_{LCRN}$.*

Proof. Let $\alpha, \beta \in [0, \infty) \cap \mathbb{R}_{LCRN}$, and let $N_1 = (S_1, R_1)$ and $N_2 = (S_2, R_2)$ be CRNs that testify to this by computing α and β with species X and Y, respectively. We also let $\mathbf{z}_1 \in [0, \infty)^{S_1}$ and $\mathbf{z}_2 \in [0, \infty)^{S_2}$ be the equilibrium points that N_1 and N_2 use to compute α and β, i.e., $\mathbf{z}_1(X) = \alpha$ and $\mathbf{z}_2(Y) = \beta$.

Let $N = (S, R)$ be the CRN defined by $S = S_1 \cup S_2 \cup S_3$ and $R = R_1 \cup R_2 \cup R_3$ where $S_3 = \{U\}$ and R_3 consists of the two reactions

$$X + Y \xrightarrow{1} X + Y + U$$

$$U \xrightarrow{1} \emptyset.$$

Note that the ODE for U is

$$\frac{du}{dt} = xy - u,$$

and the solution for $u(t)$ with all species initialized to zero is

$$u(t) = ce^{-t} + e^{-t}\int_1^t e^s x(s)y(s)ds,$$

for some constant c. Hence,

$$\lim_{t\to\infty} u(t) = 0 + \lim_{t\to\infty} \frac{\int_1^t e^s x(s)y(s)ds}{e^t}.$$

By L'Hôpital's rule, we have

$$\lim_{t\to\infty} u(t) = \lim_{t\to\infty} x(t)y(t) = \alpha\beta.$$

It remains to be shown that the equilibrium point $(\mathbf{z}_1, \mathbf{z}_2, \alpha\beta)$ is exponentially stable. First, we fix an order S_1, S_2, S_3 of the species in the Jacobian matrix J_N. We use $J_{i,j}$ to denote submatrix of J_N that contains the partial derivatives of each species $A \in S_i$ with respect to each species $B \in S_j$ for $i, j \in \{1, 2, 3\}$. Then J_N can be written as follows.

$$J_N = \begin{pmatrix} J_{N_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & J_{N_2} & \mathbf{0} \\ J_{3,1} & J_{3,2} & J_{3,3} \end{pmatrix}$$

Since N_1 and N_2 have disjoint species, it is clear that $J_{1,2} = \mathbf{0}$ and $J_{2,1} = \mathbf{0}$. Furthermore, N_1 and N_2 are unaffected by the species Y, so $J_{1,3} = \mathbf{0}$ and $J_{2,3} = \mathbf{0}$. We also note that $J_{3,3}$ contains one element $\frac{\partial}{\partial u}(xy - u) = -1$.

Since J_N is a lower triangular block matrix,

$$|J - \lambda I| = |J_{N_1} - \lambda I_1| \ \cdot \ |J_{N_2} - \lambda I_2| \ \cdot \ |-1 - \lambda|. \tag{2.4.5}$$

We can now conclude the eigenvalues of J_N are

1) the eigenvalues of J_{N_1},

2) the eigenvalues of J_{N_2}, and

3) the eigenvalue of $J_{3,3}$, which is -1.

Since α, β are both in $\mathbb{R}_{LCRN}$, the real part of the eigenvalues of J_{N_1} and J_{N_2} are all negative. Thus all the eigenvalues of J_N have negative real parts. Therefore, $\alpha\beta \in \mathbb{R}_{LCRN}$. □

Corollary 2.4.8. *$\mathbb{R}_{LCRN} \setminus \{0\}$ is a multiplicative subgroup of $\mathbb{R} \setminus \{0\}$.*

Proof. Lemma 2.4.2 tells us that $1 \in \mathbb{R}_{LCRN}$, so this follows immediately, from Lemmas 2.4.4 and 2.4.7. □

We now have the main result of this section.

Theorem 2.4.9. *$\mathbb{R}_{LCRN}$ is a subfield of $\mathbb{R}$.*

Proof. This follows immediately from Corollaries 2.4.6 and 2.4.8. □

2.5 Algebraic Numbers Are Lyapunov CRN-Computable

In this section, we prove that every algebraic number is Lyapunov CRN-computable. We begin by proving that algebraic numbers that are the smallest positive root of a polynomial with distinct roots are Lyapunov CRN-computable. In this case, we construct a CRN with one species that when initialized to zero asymptotically approaches the smallest positive root of the polynomial. We also ensure that the root is an exponentially stable equilibrium point of the CRN.

For an arbitrary algebraic number, we reduce the problem to the special case by shifting all the roots of its minimal polynomial by a rational number. By doing so, we make the relevant root become the smallest positive root, and use the special case to complete the proof of the theorem.

Lemma 2.5.1. *Every algebraic number that is the smallest positive root of some integral polynomial with roots only of multiplicity one is in $\mathbb{R}_{LCRN}$.*

Proof. Let $\alpha > 0$ be an algebraic number, and let $P(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$ be the polynomial with integral coefficients that testifies to this. Furthermore, we assume that α is the first positive root of P and that the roots of P only have multiplicity one. Without loss of generality, we

also assume that $c_0 \geq 0$, and use $-P(x)$ otherwise. We now build a CRN $N = (S, R)$ with species $S = \{X\}$ such that

$$\frac{dx}{dt} = P(x).$$

For each term $c_k x^k$ in $P(x)$, we include a reaction in R to add this term to $\frac{dx}{dt}$. If $c_k > 0$, we add the reaction

$$kX \xrightarrow{c_k} (k+1)X,$$

and if $c_k < 0$ we add the reaction

$$kX \xrightarrow{-c_k} (k-1)X.$$

Note that it is possible for the number of products or reactants to be empty. For example, the reaction for c_0 is $\emptyset \xrightarrow{c_0} X$ because $c_0 > 0$. Similarly, if $c_1 < 0$, the reaction would be $X \xrightarrow{-c_1} \emptyset$. Also, if $c_k = 0$, we do not add any reaction.

It now suffices to show that α is an exponentially stable equilibrium point of N and that if X is initialized to zero, then $x(t)$ converges to α. Since $c_0 > 0$, it is clear that $P(0) > 0$, whence $\frac{dx}{dt} > 0$ at time 0. Finally, since α is the first positive root of $P(x)$ and $x(0) = 0$, it is clear that $\lim_{t\to\infty} x(t) = \alpha$.

To show that α is exponentially stable, it suffices to show that all eigenvalues of the Jacobian matrix at α have negative real parts, i.e., that $P'(\alpha) < 0$. We show this using the following two facts.

1. $P(x) > 0$ for all $x \in [0, \alpha)$. Otherwise α would not be the smallest positive root.
2. $P'(\alpha) \neq 0$. Otherwise α would have multiplicity of at least two, but we assumed the roots of P have multiplicity one.

Finally, it is clear that

$$P'(\alpha) = \lim_{x\to\alpha^-} \frac{P(x) - P(\alpha)}{x - \alpha} < 0,$$

since $P(x) > 0$ for all $x \in [0, \alpha)$ and $P'(x) \neq 0$. □

Theorem 2.5.2. *Every algebraic number is an element of* $\mathbb{R}_{LCRN}$.

Proof. Let $\alpha > 0$ be an algebraic number, and let $P(x)$ be the minimal polynomial with integral coefficients that testifies to this. Since P is minimal, its roots have multiplicity one [18]. Therefore, if α is the smallest positive root of P, then $\alpha \in \mathbb{R}_{LCRN}$ by Lemma 2.5.1.

If α is not the smallest positive root of P, let β be the largest positive root less than α. Now let $\frac{p}{q}$ be a rational satisfying $\beta < \frac{p}{q} < \alpha$, and let $\gamma = \alpha - \frac{p}{q}$. If n is the degree of P, then $P\left(x + \frac{p}{q}\right) \cdot q^n$ is an integral polynomial with distinct roots and γ is its smallest positive root. By Lemma 2.5.1, $\gamma \in \mathbb{R}_{LCRN}$, and since $\mathbb{Q} \subseteq \mathbb{R}_{LCRN}$ and $\mathbb{R}_{LCRN}$ is closed under addition, $\frac{p}{q} + \gamma = \alpha \in \mathbb{R}_{LCRN}$. □

2.6 Discussion

We have shown that

$$\textsc{Alg} \subseteq \mathbb{R}_{LCRN} \subseteq \mathbb{R}_{RTCRN}, \tag{2.6.1}$$

where Alg is the field of algebraic numbers. We will show that

$$\textsc{Alg} \neq \mathbb{R}_{RTCRN},$$

in Chapter 3 by constructing CRNs to compute transcendental numbers. At the time of this writing we do not know whether the left-hand inclusion is proper, and we do not know whether the right-hand inclusion is proper.

Both notions of real time CRN computability discussed here, $\mathbb{R}_{RTCRN}$ and $\mathbb{R}_{LCRN}$, are closely related to the investigations by Bournez, Fraigniaud, and Koegler [3, 30] of computability by large population protocols. Roughly speaking, a *large population protocol* (*LPP*) is a deterministic chemical reaction network in which every reaction has exactly two reactants and exactly two products. Among other things, this implies that the sum of concentrations of all species is constant over time. A real number α is defined to be *computable by an LPP* if there exist an LPP N, a state $\mathbf{z}$ of N, and a designated subset D of the species of N with the following three properties.

(1) N has only finitely many fixed points.

(2) $\mathbf{z}$ is an exponentially stable state of N.

(3) α is the sum of the concentrations of the species in D in the state $\mathbf{z}$.

Bournez, Fraigniaud, and Koegler prove that a real number α is computable by an LPP if and only if α is an algebraic number. The "only if" direction of their proof is an elimination of quantifiers argument [42] that depends crucially on (1) above. It is to be hoped that further research will clarify the relationship between LPP computability and real time CRN computability.

What does (2.6.1) say about the complexity of algebraic irrationals on other models of computation?

The first thing to understand here is that deterministic chemical reaction networks are, in a very precise sense, a model of analog computation. In 1941, Shannon [51] introduced the *general-purpose analog computer* (*GPAC*). A GPAC is a mathematical abstraction of the *differential analyzer*, an early analog computer that Bush [6] had constructed at MIT, and which Shannon had operated as a graduate research assistant. The GPAC model has been corrected and otherwise modified a number of times over the years [46, 36, 20, 21]. Its present form can be characterized in terms of circuits, but it is more simply characterized as a system

$$\mathbf{y}'(t) = \mathbf{p}(t, y), \tag{2.6.2}$$

of ordinary differential equations, where $\mathbf{p}$ is a vector of polynomials. A deterministic CRN is thus a special type of GPAC of the form

$$\mathbf{y}'(t) = \mathbf{p}(y), \tag{2.6.3}$$

where each component p_i of $\mathbf{p}$ has the "kinetic" form $p_i(\mathbf{y}) = q_i(\mathbf{y}) - y_i r_i(\mathbf{y})$, with q_i and r_i having nonnegative coefficients [23]. Our CRNs in this chapter have the added constraints that all the coefficients in these polynomials are integers, and all concentrations are initialized to zero. Our main theorem thus implies that all algebraic numbers are real time computable by GPACs that have only finite information coded into their parameters and initializations.

We now turn from analog computation to discrete computation. A famous conjecture of Hartmanis and Stearns [24] says that no irrational algebraic number is real time computable by a Turing machine. This conjecture has been open for over 50 years. Fischer, Meyer, and Rosenberg [14]

proved that real-time computability on a Turing machine is equivalent to linear-time computability on a Turing machine. Hence the Hartmanis-Stearns conjecture is equivalent to the statement that no irrational algebraic number is linear-time computable by a Turing machine. As observed by Gurevich and Shelah [22], linear time is a very model-dependent notion. Hence, as stated, the Hartmanis-Stearns conjecture is a very specific conjecture about linear-time computation on Turing machines.

Our main theorem does not disprove the Hartmanis-Stearns conjecture (nor was it intended to), but conceptually locating the gap between our main theorem and a disproof of the Hartmanis-Stearns conjecture would shed light on the computational complexities of algebraic irrationals. This raises the following questions.

Question 1. Can CRNs in our model (or GPACs with only finite information encoded into their parameters and initializations) produce in linear time the individual digits of each real number that is real time CRN-computable? If so, our main theorem implies that the Hartmanis-Stearns conjecture fails for analog computation. If not, the Hartmanis-Stearns conjecture holds for analog computation and is essentially about producing the individual digits as opposed to the analog convergence that we have used here.

Question 2. Is there a reasonable discrete model of computation on which some algebraic irrational can be computed in linear time? If so, then the Hartmanis-Stearns conjecture is either false or model-dependent. If not, then the Hartmanis-Stearns conjecture is true in a strong, model-independent way, at least for discrete computation. (Note that "reasonable" here excludes models that perform numerical operations faster than we know how to do them, because Brent [5] has shown how to compute $\sqrt{2}$ in linear time if integer multiplication can be done in linear time. See also [37].)

CHAPTER 3. REAL-TIME EQUIVALENCE OF CRNS AND ANALOG COMPUTERS

3.1 Introduction

In the last chapter we defined the class $\mathbb{R}_{RTCRN}$ of real time computable real numbers by chemical reaction networks. In this chapter, we investigate the relationship between real time computable real numbers by CRNs and general purpose analog computers (GPACs).

We define the class $\mathbb{R}_{RTGPAC}$ of real time computable real numbers by GPACs. Roughly, $\alpha \in \mathbb{R}_{RTGPAC}$ if there exists a polynomial initial value problem (PIVP) with integer coefficients such that, if initialized with all zeros, then all variables are bounded and one of the variables converges to α exponentially quickly. These restrictions are analogous to the definition of $\mathbb{R}_{RTCRN}$ and ensure that the PIVP is finitely specifiable and α is computed in real time. We show that $\mathbb{R}_{RTGPAC} = \mathbb{R}_{RTCRN}$ by proving that $\mathbb{R}_{RTCRN}$ is a subfield of $\mathbb{R}$ and using an extension of the difference representation introduced in [12] that relies on these closure properties. We also show that the constraint of all zero initial conditions can be relaxed to integral initial conditions. With these new theorems, we prove two well-known transcendental numbers e, π and γ, together with a less-known constant, Dottie number, are members of $\mathbb{R}_{RTCRN}$. The proofs and constructions for these transcendental numbers are short and concise, and demonstrate the power of these theorems for generating and proving real-time CRNs correct.

The rest of this chapter is organized as follows. Section 3.2 includes the main theorem of the chapter, that $\mathbb{R}_{RTGPAC} = \mathbb{R}_{RTCRN}$, along with the proof that $\mathbb{R}_{RTCRN}$ is a field; Section 3.3 includes proofs that e, π, γ, and Dottie number are real time computable by chemical reaction networks using the theorems from Section 3.2; after that we demonstrate some simulation result in Section 3.4.

3.2 Real-Time Equivalence of CRNs and GPACs

This section is devoted to proving that the class $\mathbb{R}_{RTCRN}$ is equivalent to an analogous class $\mathbb{R}_{RTGPAC}$ of real time computable real numbers by general purpose analog computers. We begin by formally defining $\mathbb{R}_{RTGPAC}$.

For a PIVP $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ satisfying $\mathbf{y}(0) = \mathbf{0}$, we say that $\mathbf{y}$ is an *computer* for a real number α if the following three properties hold:

1. All coefficients of $\mathbf{y}$ are integers,
2. There is a constant $\beta > 0$ such that $|y_i(t)| \leq \beta$ for all $1 \leq i \leq n$ and $t \in [0, \infty)$, and
3. $|y_1(t) - \alpha| \leq 2^{-t}$ for all $t \in [1, \infty)$.

The real numbers for which there is a computer $\mathbf{y}$ are called *real-time GPAC computable*. The set of real-time CRN computable real numbers is denoted by $\mathbb{R}_{RTGPAC}$.

Note that the constraints above mirror the definition of $\mathbb{R}_{RTCRN}$ except for the fact that $y_1(t)$ is converging to α instead of $|\alpha|$. This difference is due to the CRN restriction of species concentrations to be non-negative real numbers whereas the value of a GPAC variable $y_i(t)$ has no such restriction.

Lemma 3.2.1. $\mathbb{R}_{RTCRN} \subseteq \mathbb{R}_{RTGPAC}$.

Proof. Given a computer (N, Y_1) for $\alpha \in \mathbb{R}$, let $\mathbf{y}$ be the PIVP induced by the deterministic semantics of N. Note that (N, Y_1) computes α when its species concentrations are initialized to zero, therefore $\mathbf{y}(0) = \mathbf{0}$. The fact that $\mathbf{y}$ is also a computer for α immediately follows from the constraints imposed on N and the fact that if $\alpha < 0$, a multiplying each ODE by -1 causes $y_1(t)$ to converge directly to α instead of $|\alpha|$. □

Although the inclusion above is trivial, the fact that $\mathbb{R}_{RTGPAC} \subseteq \mathbb{R}_{RTCRN}$ is not so obvious. This is due to deterministic CRNs inducing PIVPs with restricted forms, namely, the polynomial of each ODE has the structure

$$y'(t) = p(t) - q(t)y(t),$$

where p and q are polynomials over the concentrations of the species. The fact that negative terms in the ODE for Y must depend on its own concentration $y(t)$ makes certain GPAC constructions difficult to implement with CRNs.

The rest of this section is devoted to finishing the proof of the main theorem: $\mathbb{R}_{RTGPAC} = \mathbb{R}_{RTCRN}$. To simplify the proof, we first prove that $\mathbb{R}_{RTCRN}$ is a subfield of $\mathbb{R}$ which solves an open problem stated in [26]. The proofs of closure under addition, multiplication, division, and subtraction rely on certain convergence properties. Thus, we first state and prove two lemmas which demonstrate that certain differential equations immediately yield exponential convergence to a target real number. Then we prove the four closure properties necessary to show that $\mathbb{R}_{RTCRN}$ is a field using these lemmas. Finally, we conclude with the proof of the main theorem that $\mathbb{R}_{RTGPAC} = \mathbb{R}_{RTCRN}$.

Lemma 3.2.2 (Direct Convergence Lemma)**.** *If $\alpha \in \mathbb{R}$ and $x, f : [0, \infty) \to \mathbb{R}$ are functions that satisfy*

$$x'(t) = f(t) - x(t) \text{ for all } t \in [0, \infty) \tag{3.2.1}$$

$$|f(t) - \alpha| \leq e^{-t} \text{ for all } t \in [1, \infty), \tag{3.2.2}$$

then there exist constants $\gamma, \tau \in (0, \infty)$ such that

$$|x(t) - \alpha| \leq e^{-\gamma t} \text{ for all } t \in [\tau, \infty). \tag{3.2.3}$$

Proof. Assume the hypothesis. The ODE of eq. (3.2.1) can be solved directly using the integrating factor method and has a solution of the form

$$x(t) = e^{-t} \int e^t f(t) dt. \tag{3.2.4}$$

By eq. (3.2.2), we know that for all $t \geq 1$,

$$\int e^t f(t) dt \leq \int e^t \left(\alpha + e^{-t}\right) dt = \alpha e^t + t + C_1,$$

for some constant C_1. This, along with eq. (3.2.4), yields

$$x(t) \leq \alpha + e^{-t} (t + C_1). \tag{3.2.5}$$

Using a similar argument, it is easy to show that

$$x(t) \geq \alpha - e^{-t}\,(t + C_2) \tag{3.2.6}$$

for some constant C_2. Choosing $C = \max\{\,0, C_1, C_2\,\}$, it follows from eqs. (3.2.5) and (3.2.6) that

$$|x(t) - \alpha| \leq (t + C)e^{-t} \leq e^{-t/2},$$

for all $t \geq \max\{1, 4\log(C+1)\}$. □

Lemma 3.2.3 (Reciprocal Convergence Lemma)**.** *If $\alpha \in \mathbb{R}_{>0}$ and $x, f : [0,\infty) \to \mathbb{R}$ are continuous functions that satisfy*

$$x'(t) = 1 - f(t) \cdot x(t) \text{ for all } t \in [0,\infty) \tag{3.2.7}$$

$$|f(t) - \alpha| \leq e^{-t} \text{ for all } t \in [1,\infty), \tag{3.2.8}$$

then there exist constants $\gamma, \tau > 0$ such that

$$\left|x - \frac{1}{\alpha}\right| \leq e^{-\gamma t} \text{ for all } t \in [\tau,\infty). \tag{3.2.9}$$

Proof. Assume the hypothesis. Since f is continuous, its antiderivative exists, so the ODE from eq. (3.2.7) can be solved directly using the integrating factor method with a solution of the form

$$x(t) = e^{-F(t)} \int_0^t e^{F(s)} ds, \tag{3.2.10}$$

where $F(t) = \int_0^t f(s)ds$. If we let $h(t) = f(t) - \alpha$, and let $H(t) = \int_0^t h(s)ds$ be the antiderivative of h, then

$$F(t) = \int_0^t (\alpha + h(s))\, ds = \alpha t + H(t).$$

Using this relationship, we can rewrite eq. (3.2.10) as

$$x(t) = e^{-F(t)} \cdot \frac{1}{\alpha} \int_0^t e^{H(s)}\,(\alpha e^{\alpha s})\, ds. \tag{3.2.11}$$

We can now use integration by parts on the integral of eq. (3.2.11) with $u(s) = e^{H(s)}$ and $v'(s) = \alpha e^{\alpha s}$ to obtain

$$\begin{aligned}\int_0^t e^{H(s)}\left(\alpha e^{\alpha s}\right) ds &= \int_0^t u(s)v'(s)ds \\ &= u(s)v(s)\Big|_0^t - \int_0^t v(s)u'(s)ds \\ &= e^{H(t)}e^{\alpha t} - 1 - \int_0^t e^{\alpha s}\left(h(s)e^{H(s)}\right) ds.\end{aligned}$$

Substituting this into eq. (3.2.11) and using the fact that $F(t) = \alpha t + H(t)$, we obtain

$$x(t) = e^{-F(t)} \cdot \frac{1}{\alpha}\left(e^{F(t)} - 1 - \int_0^t h(s)e^{F(s)}ds\right),$$

which yields the following bound:

$$\left|x(t) - \frac{1}{\alpha}\right| \leq e^{-F(t)}\left(1 + \int_0^t |h(s)|e^{F(s)}ds\right). \tag{3.2.12}$$

It remains to be shown that the right-hand side of eq. (3.2.12) is bounded by an exponential after some time τ. We begin by showing that $H(t)$ is bounded above and below by the constant $C_1 = \int_0^1 |h(s)|ds + \frac{1}{e}$:

$$|H(t)| \leq \int_0^t |h(s)|ds \leq \int_0^1 |h(s)|ds + \int_1^t e^{-s}ds = C_1 - e^{-t} \leq C_1.$$

It immediately follows that

$$\begin{aligned}e^{F(t)} &= e^{\alpha t + H(t)} \leq C_2 e^{\alpha t} \\ e^{-F(t)} &= e^{-\alpha t - H(t)} \leq C_2 e^{-\alpha t}\end{aligned}$$

where $C_2 = e^{C_1}$. If we define the constant $C_3 = \int_0^1 |h(s)|e^{F(s)}ds$, we can bound the integral of eq. (3.2.12) with

$$\int_0^t |h(s)|e^{F(s)}ds \leq C_3 + \int_1^t e^{-s}\left(C_2 e^{\alpha s}\right) ds = C_5 + C_4 e^{(\alpha-1)t}$$

where $C_4 = \frac{C_2}{\alpha - 1}$ and $C_5 = C_3 - C_4 e^{\alpha - 1}$. Thus, we can rewrite eq. (3.2.12):

$$\left|x(t) - \frac{1}{\alpha}\right| \leq C_2 e^{-\alpha t}\left(1 + C_5 + C_4 e^{(\alpha-1)t}\right) = C_6 e^{-\alpha t} + C_7 e^{-t}$$

where $C_6 = C_2(1 + C_5)$ and $C_7 = C_2C_4$.

It immediately follows that there exist constants γ and τ such that $\left|x(t) - \frac{1}{a}\right|$ is bounded by $e^{-\gamma t}$ for all $t \in [\tau, \infty)$. □

Using lemmas 3.2.2 and 3.2.3, we now prove that $\mathbb{R}_{RTCRN}$ is a field. We split the four closure properties into the following four lemmas.

Notation 3.2.4. *Two CRNs can naturally be combined into one. Given $N_1 = (S_1, R_1)$ and $N_2 = (S_2, R_2)$, we define the join of N_1 and N_2 to be the CRN*

$$N_1 \sqcup N_2 = (S_1 \cup S_2, R_1 \cup R_2). \tag{3.2.13}$$

Lemma 3.2.5. *If $\alpha, \beta \in \mathbb{R}_{RTCRN}$, then $\alpha + \beta \in \mathbb{R}_{RTCRN}$.*

Proof. Assume the hypothesis, and let (N_α, X) and (N_β, Y) be CRN computers that compute α and β, respectively. Without loss of generality, we assume that $\alpha, \beta \geq 0$ and that N_α and N_β do not share any species.

Now let Z be a new species, and let $N = N_\alpha \sqcup N_\beta \sqcup \widehat{N}$ where $\widehat{N}$ is the CRN defined by the reactions

$$X \xrightarrow{1} X + Z$$
$$Y \xrightarrow{1} Y + Z$$
$$Z \xrightarrow{1} \emptyset.$$

Note that the species in N_α and N_β are unaffected by the reactions of $\widehat{N}$, and the ODE for Z is:

$$z'(t) = x(t) + y(t) - z(t). \tag{3.2.14}$$

Let $f(t) = x(t) + y(t)$. By lemma 2.3.1, without loss of generality, we can assume that $|f(t) - \alpha - \beta| \leq e^{-t}$ for all $t \geq 1$. Immediately by lemmas 2.3.1 and 3.2.2, we conclude that $\alpha + \beta \in \mathbb{R}_{RTCRN}$. □

Lemma 3.2.6. *If $\alpha, \beta \in \mathbb{R}_{RTCRN}$, then $\alpha\beta \in \mathbb{R}_{RTCRN}$.*

Proof. Assume the hypothesis, and let (N_α, X) and (N_β, Y) be CRN computers that compute α and β, respectively Furthermore, we assume that N_α and N_β do not share any species. Without loss of generality, we also assume that $\alpha, \beta \geq 0$.

Now let Z be a new species, and let $N = N_\alpha \sqcup N_\beta \sqcup \widehat{N}$ where $\widehat{N}$ is the CRN defined by the reactions

$$X + Y \xrightarrow{1} X + Y + Z$$
$$Z \xrightarrow{1} \emptyset.$$

Note that the species in N_α and N_β are unaffected by the reactions of $\widehat{N}$ and yields the following ODE for Z:

$$z'(t) = x(t)y(t) - z(t). \tag{3.2.15}$$

Let $f(t) = x(t)y(t)$. By lemma 2.3.1, without out loss of generality, we can assume that $|f(t) - \alpha\beta| \leq e^{-t}$ for all $t \geq 1$. Immediately by lemmas 2.3.1 and 3.2.2, we conclude that $\alpha\beta \in \mathbb{R}_{RTCRN}$. □

Lemma 3.2.7. *If $\alpha \in \mathbb{R}_{RTCRN}$ and $\alpha \neq 0$, then $\frac{1}{\alpha} \in \mathbb{R}_{RTCRN}$.*

Proof. Assume the hypothesis, and let (N_α, X) be CRN a computer that testifies to this. Without loss of generality, we also assume that $\alpha > 0$.

Now let Y be a new species, and let $N = N_\alpha \sqcup \widehat{N}$ where $\widehat{N}$ is the CRN defined by the reactions

$$\emptyset \xrightarrow{1} Y$$
$$X + Y \xrightarrow{1} X.$$

Note that the species in N_α are unaffected by the reactions of $\widehat{N}$ and yields the following ODE for Y:

$$z'(t) = 1 - x(t)y(t). \tag{3.2.16}$$

Since $\alpha \in \mathbb{R}_{RTCRN}$, we know that $|f(t) - \alpha| \leq e^{-t}$ for all $t \geq 1$. It follows from lemmas 2.3.1 and 3.2.3 that $\frac{1}{\alpha} \in \mathbb{R}_{RTCRN}$.

□

Lemma 3.2.8. *If $\alpha, \beta \in \mathbb{R}_{RTCRN}$, then $\alpha - \beta \in \mathbb{R}_{RTCRN}$.*

Proof. Assume the hypothesis, and let (N_α, X) and (N_β, Y) be CRN computers that compute α and β, respectively. Furthermore, we assume that N_α and N_β do not share any species. Without loss of generality, we also assume that $\alpha > \beta \geq 0$.

Now let Z be a new species, and let $N = N_\alpha \sqcup N_\beta \sqcup \widehat{N}$ where $\widehat{N}$ is the CRN defined by the reactions

$$\emptyset \xrightarrow{1} Z$$
$$X + Z \xrightarrow{1} X$$
$$Y + Z \xrightarrow{1} Y + 2Z.$$

Note that the species in N_α and N_β are unaffected by the reactions of $\widehat{N}$ and yields the following ODE for Z:

$$z'(t) = 1 - (x(t) - y(t))z(t). \tag{3.2.17}$$

Let $f(t) = x(t) - y(t)$. By lemma 2.3.1, without out loss of generality, we can assume that $|f(t) - (\alpha - \beta)| \leq e^{-t}$ for all $t \geq 1$. By lemmas 2.3.1 and 3.2.3, we know that $\frac{1}{\alpha - \beta} \in \mathbb{R}_{RTCRN}$. By lemma 3.2.7, we conclude that $\alpha - \beta \in \mathbb{R}_{RTCRN}$. □

Theorem 3.2.9. *$\mathbb{R}_{RTCRN}$ is a subfield of $\mathbb{R}$.*

Proof. This immediately follows from Lemmas 3.2.5–3.2.8 and the fact that $\mathbb{R}_{RTCRN}$ is non-empty. □

As a consequence of Theorem 3.2.9, and the results of [26] we now know that $\mathbb{R}_{RTCRN}$ contains all algebraic numbers and an infinite family of transcendental numbers. However, we have yet to prove that natural transcendentals such as e and π are real-time computable by CRNs. These proofs are simplified dramatically using the following theorem which uses a construction similar to [12].

Theorem 3.2.10. *$\mathbb{R}_{RTCRN} = \mathbb{R}_{RTGPAC}$.*

Proof. We have already shown the forward direction in Lemma 3.2.1.

For the backward direction, assume that $0 \neq \alpha \in \mathbb{R}_{RTGPAC}$, and let $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ be the PIVP that testifies to this. Then the individual components of $\mathbf{y}$ obey the ODEs

$$\begin{aligned} y_1' &= p_1(y_1, \ldots, y_n), \\ y_2' &= p_2(y_1, \ldots, y_n), \\ &\vdots \\ y_n' &= p_n(y_1, \ldots, y_n). \end{aligned}$$

For each $1 \leq i \leq n$, we define the variables $\hat{\mathbf{y}} = (z, u_1, v_1, u_2, v_2, \ldots, u_n, v_n)$ as well as the polynomials

$$\hat{p}_i(\hat{\mathbf{y}}) = p_i(u_1 - v_1, u_2 - v_2, \ldots, u_n - v_n),$$

noting that each $\hat{p}_i$ is indeed an integral polynomial over the variables of $\hat{\mathbf{y}}$. For each $1 \leq i \leq n$, we also define the polynomials $\hat{p}_i^+$ and $\hat{p}_i^-$ by the positive and negative terms of $\hat{p}_i$, respectively, whence $\hat{p}_i = \hat{p}_i^+ - \hat{p}_i^-$.

We now define ODEs for each variable u_i and v_i of $\hat{\mathbf{y}}$,

$$u_i' = \hat{p}_i^+ - u_i v_i \left(\hat{p}_i^+ + \hat{p}_i^-\right), \tag{3.2.18}$$

$$v_i' = \hat{p}_i^- - u_i v_i \left(\hat{p}_i^+ + \hat{p}_i^-\right), \tag{3.2.19}$$

as well as the ODE for the variable z

$$z' = 1 - (u_1 - v_1)z. \tag{3.2.20}$$

Notice that if $y_i = u_i - v_i$, then

$$u_i' - v_i' = \hat{p}_i^+ - \hat{p}_i^- = \hat{p}_i = p_i = y_i',$$

therefore if $\hat{\mathbf{y}}(0) = \mathbf{0}$, we know that $y_i(t) = u_i(t) - v_i(t)$ for all $t \in [0, \infty)$.

We now prove that every variable of $\hat{\mathbf{y}}$ is bounded from above by some constant. For the sake of contradiction, assume that either u_i or v_i is unbounded. Recall that each variable of $\mathbf{y}$ is bounded by some $\beta > 0$, and therefore $-\beta \leq y_i(t) \leq \beta$ for all $t \in [0, \infty)$. Since $y_i(t) = u_i(t) - v_i(t)$, it follows that *both* u_i and v_i must be unbounded. However, this is a contradiction since u_i' and v_i' each include

the negative terms $-u_i v_i \left(\hat{p}_i^+ + \hat{p}_i^-\right)$ which grow faster than their positive terms. Thus, u_i and v_i must both be bounded.

Since each of the ODEs of $\hat{\mathbf{y}}$ can be written in the form $x' = p - qx$ where p and q are polynomials with positive integral coefficients, there exists a CRN $N = (S, R)$ with species $S = \{U_i, V_i \mid 1 \leq i \leq n\} \cup \{Z\}$ that obey these ODEs. Because $y_1 = u_1 - v_1$, this means that $|u_1(t) - v_1(t) - \alpha| \leq 2^{-t}$. By Lemma 3.2.3, it immediately follows that N real time computes $\frac{1}{\alpha}$ with species Z. Finally, we obtain that $\alpha \in \mathbb{R}_{RTCRN}$ by closure under reciprocal. □

3.3 A Tale of Four Numbers

In this section, we will prove that e, π, γ, and Dottie number are real time computable by CRNs. However, first we prove a useful theorem that shows that the constraint that the CRN or GPAC must be initialized to all zeros can be relaxed to any integral initial condition.

Theorem 3.3.1. *If $\alpha \in \mathbb{R}$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, $\mathbf{y}(0) \in \mathbb{Z}^n$ is a PIVP such that*

1. *$|y_i(t)| \leq \beta$ for all $1 \leq i \leq n$ and $t \in [0, \infty)$ for some $\beta > 0$, and*
2. *$|y_1(t) - \alpha| \leq 2^{-t}$ for all $t \in [0, \infty)$,*

then $\alpha \in \mathbb{R}_{RTGPAC}$.

Proof. Assume the hypothesis. Then there is a polynomial p_i corresponding to each variable y_i of $\mathbf{y}$ such that $y_i' = p_i$. We will now define a related PIVP that when initialized to all zeros computes α.

Define the variables $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)$ that obey the ODEs

$$\hat{y}_i' = p_i(\hat{y}_1 + y_1(0), \hat{y}_2 + y_2(0), \ldots, \hat{y}_n + y_n(0)).$$

Since $\mathbf{y}(0) \in \mathbb{Z}^n$, each ODE $\hat{y}_i$ is a polynomial with integral coefficients. We also note that if $\hat{y}_i(t) = y_i(t) - y_i(0)$ for some $t \in [0, \infty)$, then

$$\hat{y}_i'(t) = p_i(y_1(t), y_2(t), \ldots, y_n(t)) = y_i'(t).$$

Thus, if we initialize $\hat{\mathbf{y}}(0) = \mathbf{0}$, it follows that $\hat{y}_i(t) = y_i(t) - y_i(0)$ for all $t \in [0, \infty)$. Since the PIVP $\mathbf{y}$ computes α, it follows that the PIVP $\hat{\mathbf{y}}$ computes $\alpha - y_1(0)$, and therefore $\alpha - y_1(0) \in \mathbb{R}_{RTGPAC}$.

Finally, since $y_1(0) \in \mathbb{Z}$, it is also in $\mathbb{R}_{RTGPAC}$, and by closure under addition we conclude that $\alpha \in \mathbb{R}_{RTGPAC}$. □

This is means that having non-zero integer value resulting in equivalent definition of $\mathbb{R}_{RTCRN}$. This theorem dramatically simplifies the constructions. Together with Theorem 3.2.10, we can developed the following process to compute a target number α.

1. Pick a function $x(t)$ that converges to α **exponentially** fast.
2. Implement $x(t)$ by a GPAC.
3. Translate the GPAC into a CRN $\mathcal{N}$, with each species x represented by a pair of variables $x_1(t)$ and $x_2(t)$ such that $x(t) = x_1(t) - x_2(t)$ by using Theorem 3.2.10.
4. Lastly, turn $\mathcal{N}$ into $\widehat{\mathcal{N}}$, with the difference of each pair of $x_1(t)$, $x_2(t)$ being approximated by a $\widehat{x}(t)$, by using Lemma 3.2.8.

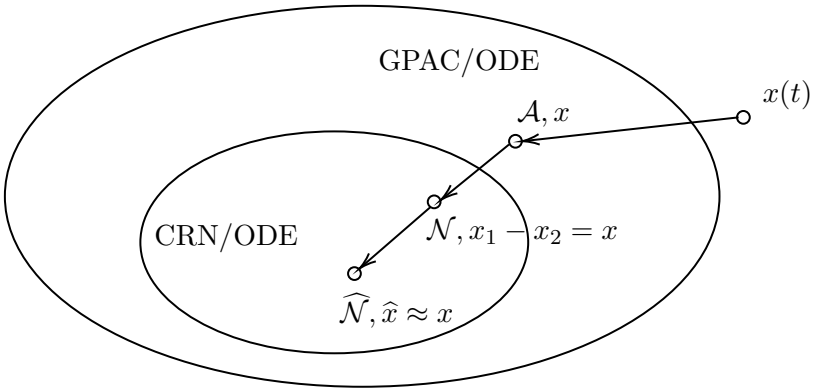

Figure 3.1: A general procedure to computer a number α.

We now present concise proofs that the e and π are members of $\mathbb{R}_{RTCRN}$.

3.3.1 e is CRN-Computable in Real Time

Theorem 3.3.2. $e \in \mathbb{R}_{RTCRN}$.

Proof. By Theorem 3.3.1, it suffices to show that there exists a CRN computer with integral initial conditions that computes e exponentially quickly. Consider the CRN defined by

$$X \xrightarrow{1} \emptyset$$
$$X + Y \xrightarrow{1} X$$

along with the initial condition $x(0) = 1$ and $y(0) = 1$. This induces the system of ODES

$$x'(t) = -x(t) \tag{3.3.1}$$
$$y'(t) = -x(t)y(t), \tag{3.3.2}$$

which is trivial to solve and has solution

$$x(t) = e^{-t}, \qquad y(t) = e^{1-e^{-t}}.$$

It is clear that $y(t)$ exponentially goes to e, and thus $e \in \mathbb{R}_{RTCRN}$.

□

It is easy to apply the construction of Theorem 3.3.1 to the CRN provided in the proof of Theorem 3.3.5, and Figure 3.2 shows the plot of this expanded CRN computing e in this way.

3.3.2 π is CRN-Computable in Real Time

Theorem 3.3.3. $\pi \in \mathbb{R}_{RTCRN}$.

Proof. By Theorem 3.3.1, it suffices to show that there exists a CRN computer with integral initial conditions that computes π exponentially quickly. Consider the CRN defined by

$$W \xrightarrow{1} \emptyset$$
$$W + X + Y \xrightarrow{2} W + Y$$
$$W + 2X \xrightarrow{1} W + 2X + Y$$
$$W + 2Y \xrightarrow{1} W + Y$$
$$W + X \xrightarrow{1} W + X + Z,$$

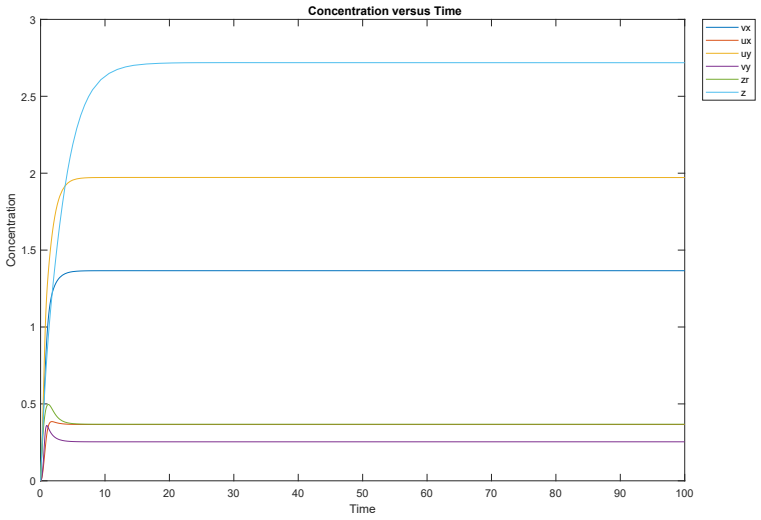

Figure 3.2: MATLAB visualization of computing e from Theorem 3.3.5. This plot is of the CRN after applying the construction from Theorem 3.3.1 so that all species concentrations are initially zero, and Z is the species converging to e.

with initial condition $w(0) = x(0) = 1$ and $y(0) = z(0) = 0$. It is easy to verify that this CRN induces the following system of ODEs

$$w'(t) = -w(t), \tag{3.3.3}$$

$$x'(t) = -2w(t)x(t)y(t), \tag{3.3.4}$$

$$y'(t) = w(t)x(t)^2 - w(t)y(t)^2, \tag{3.3.5}$$

$$z'(t) = w(t)x(t). \tag{3.3.6}$$

By examining eq. (3.3.3), it is easy to see that $w(t) = e^{-t}$, and by examining eqs. (3.3.4) to (3.3.6), we see that we can perform a change of variable from t to $u(t) = \int_0^t w(s)ds = 1 - e^{-t}$ to obtain the equivalent system of ODEs:

$$x'(u) = -2x(u)y(u),$$

$$y'(u) = x(u)^2 - y(u)^2,$$

$$z'(u) = x(u).$$

This system can be solved directly and has solution

$$x(u) = \frac{1}{u^2+1}, \qquad y(u) = \frac{u}{u^2+1}, \qquad z(u) = \arctan(u).$$

Since $\mathbb{R}_{RTCRN}$ is a field, it now suffices to show that $z(t) = z(u(t)) = \arctan(1-e^{-t})$ converges to $\frac{\pi}{4}$ exponentially quickly. Note that Taylor expansion of the function $\arctan(x)$ around 1 gives

$$\arctan(x) = \frac{\pi}{4} + \frac{x-1}{2} - \frac{1}{4}(x-1)^2 + o((x-1)^2).$$

Thus we obtain

$$\arctan(u(t)) - \frac{\pi}{4} = O(u(t)-1) = O(e^{-t}).$$

Hence $\arctan(1-e^{-t})$ converges to $\frac{\pi}{4}$ exponentially quickly, and therefore $\pi \in \mathbb{R}_{RTCRN}$. □

It is easy to generate the reactions of the explicit CRN that computes π from an all-zero initial condition. The plot of this CRN is provided in Figure 3.3.

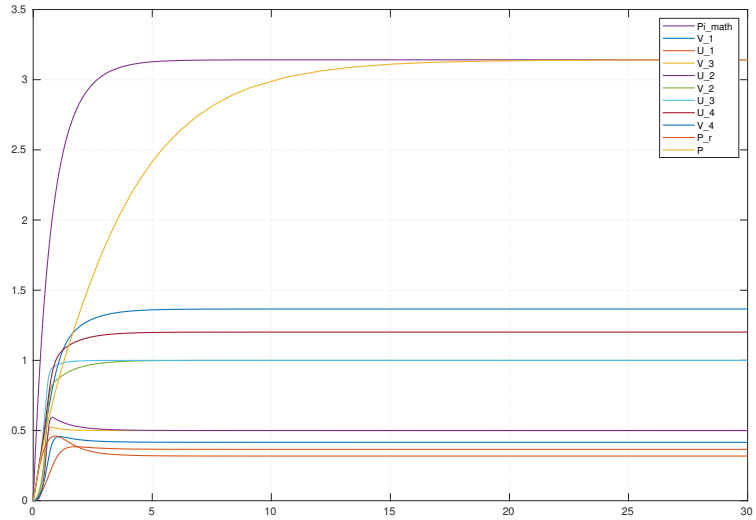

Figure 3.3: MATLAB visualization of computing π from Theorem 3.3.5. This plot is of the CRN after applying the construction from Theorem 3.3.1 so that all species concentrations are initially zero, and P is the species converging to π.

3.3.3 γ is CRN-Computable in Real Time

Euler's constant, γ, is a famous constant in mathematical anaylysis and number theory. It is defined by the limiting difference between the *harmoic series* and the *natural logarithm*.

$$\gamma = \lim_{n\to\infty} \left(\ln n - \sum_{k=1}^{n} \frac{1}{k} \right) \approx 0.57721.$$

However, the above expression converges to γ very slowly. Instead, we will rely on the following formula from [27] Equation (60)

$$\gamma = 1 - \Gamma'(2).$$

to start our construction. Note that Γ is the *Gamma function*, defined as

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} \, \mathrm{d}t.$$

Theorem 3.3.4. $\gamma \in \mathbb{R}_{RTCRN}$.

It suffices to compute $\Gamma'(2)$, for which we have

$$\begin{aligned}
\Gamma'(2) &= \int_0^\infty e^{-t} t \log(t) \, \mathrm{d}t, \\
&= \int_0^1 e^{-t} t \log(t) \, \mathrm{d}t + \int_1^\infty e^{-t} t \log(t) \, \mathrm{d}t, \\
&= \int_0^1 e^{-(1-s)} (1-s) \log(1-s) \, \mathrm{d}s + \int_0^\infty e^{-(1+s)} (1+s) \log(1+s) \, \mathrm{d}s, \\
&= e^{-1} \left(\int_0^1 e^{s} (1-s) \log(1-s) \, \mathrm{d}s + \int_0^\infty e^{-s} (1+s) \log(1+s) \, \mathrm{d}s \right),
\end{aligned}$$

where in the second to the last equation we use change of variables by letting $t = 1 - s$ in the first integral, $t = 1 + s$ in the second integral. Let

$$\alpha \equiv \int_0^1 e^{s} s \log(1-s) \, \mathrm{d}s$$

and

$$\beta \equiv \int_0^\infty e^{-s} (1+s) \log(1+s) \, \mathrm{d}s.$$

Note that both α and β are real. It suffices to show α and β are real-time computable.

We let

$$g(t) \equiv \int_0^{1-e^{-t}} e^s(1-s)\log(1-s)\,\mathrm{d}s, \qquad \text{and} \qquad f(t) \equiv \int_0^t e^{-s}(1+s)\log(1+s)\,\mathrm{d}s.$$

Note that

$$\lim_{t\to\infty} g(t) = \alpha, \qquad \lim_{t\to\infty} f(t) = \beta,$$

by the definition of α and β. We now have $f(t)$ and $g(t)$ as our target functions. Next, we will construct ODE systems that compute $f(t)$ and $g(t)$.

We start with $f(t)$. After taking the derivative of f, we have

$$f' = e^{-t}(1+t)\log(1+t)$$

Let

$$w \equiv e^{-t}(1+t)\log(1+t),$$

Then

$$w' = -w + \left(e^{-t}\log(1+t) + e^{-t}\right)$$

Let

$$u \equiv e^{-t}\log(1+t), \qquad v \equiv e^{-t}, \qquad r \equiv \frac{1}{1+t}.$$

Therefore we have

$$w' = -w + u + v,$$

and

$$v' = -v,$$

$$r' = -r^2,$$

and

$$u' = -u + rv,$$

with initial values such that $v(0) = r(0) = 1$ and $u(0) = w(0) = 0$.

As for $g(t)$, we let

$$p \equiv e^{1-e^{-t}}, \qquad q \equiv te^{-t}.$$

Then

$$\begin{aligned} g' &= e^{1-e^{-t}} e^{-t} \cdot (-t) \cdot (1-e^{-t})' \\ &= -pqv, \end{aligned}$$

and

$$p' = pv, \qquad q' = v - q,$$

with $p(0) = 1$ and $q(0) = 0$.

Putting all the above together, the following PIVP is what we need:

$$\begin{cases} f' = w, \\ g' = -pqv, \\ w' = -w + u + v, \\ u' = -u + rv, \\ v' = -v, \\ r' = -r^2, \\ p' = pv, \\ q' = v - q, \end{cases}$$

with $f(0) = g(0) = u(0) = w(0) = q(0) = 0$ and $v(0) = r(0) = p(0) = 1$.

Hence we have a GPAC that contains two variables f and g, which compute α and β respectively. The only thing left to show is the exponential rate of convergence.

First, note that $|f(t) - \beta| = O(t^2 e^{-t})$ is given by [27] Equation (65). We only need to estimate $|g(t) - \alpha|$, for which we have

$$\begin{aligned} |g(t) - \alpha| &= \left| \int_0^1 e^s (1-s) \log(1-s) \, \mathrm{d}s - \int_0^{1-e^{-t}} e^s (1-s) \log(1-s) \, \mathrm{d}s \right| \\ &= \left| \int_0^\infty e^{1-e^{-t}} e^{-2t} \cdot (-t) \, \mathrm{d}t - \int_0^t e^{1-e^{-t}} e^{-2t} \cdot (-t) \, \mathrm{d}t \right|, \text{by change of variable } s = 1 - e^{-t} \\ &= \int_t^\infty e^{1-e^{-t}} e^{-2t} t \, \mathrm{d}t \leq e \int_t^\infty e^{-2t} t \, \mathrm{d}t \end{aligned}$$

Integrate by parts, we can see

$$\int_t^\infty e^{-2t} t \, \mathrm{d}t = O(te^{-2t}).$$

So α, β can be computed by CRN in real time by Theorem 3.2.10. Then by Lemma 3.2.8, γ can be computed by CRN in real time.

3.3.4 Dottie Number is CRN-Computable in Real Time

In the previous sections, when we computed e, π, or γ, we had some more or less explicit functions from other sources, which we know converge exponentially fast to a target number. What if we don't have the luxury of having such ready-to-use functions? We will show one example related to the so-called Dottie number.

The *Dottie number* is a mathematical constant defined by the unique real root of the equation

$$\cos x = x,$$

whose approximate value is 0.739085... in radians. It is the only real fixed point of the cosine function, and is a nontrivial example of a universal attracting fixed point. That is,

$$\lim_{n\to\infty} \cos^n(x_0) = \text{Dottie number}$$

for any $x_0 \in \mathbb{R}$.

Theorem 3.3.5. *Dottie number is in $\mathbb{R}_{RTCRN}$.*

Proof. The major idea is to make some $y(t)$ such that

$$y' = \cos(y) - y.$$

Note that the right hand side of the above ODE has negative eigenvalue at the Dottie number. Therefore, y converges exponentially fast to it.

We only need to convert the system into a PIVP. Let $u = \cos(y)$ and $v = \sin(y)$. Consider their derivatives, we have

$$\begin{cases} y' = u - y, \\ u' = -vy' = -v(u - y), \\ v' = uy' = u(u - y), \end{cases}$$

with $y(0) = 0$, $u(0) = 1$, and $v(0) = 0$. Therefore, by Theorem 3.2.10, Dottie number is in $\mathbb{R}_{RTCRN}$. □

3.4 Experimental Results

In this section, we walk through the whole construction and simulation process of computing e and π by CRNs. We skip the case for γ and Dottie number, since the process is more or less the same.

3.4.1 Construction and Simulation of e

In the proof of Theorem 3.3.5, we used a CRN with integer initial conditions to compute e. In this section, we apply the construction of Theorem 3.3.1 to generate a CRN that computes e from all zero initial conditions.

Recall that the PIVP used induced in Theorem 3.3.5 is

$$\frac{d\hat{x}}{dt} = -\hat{x} \qquad \frac{d\hat{y}}{dt} = -\hat{x}\hat{y},$$

with initial condition $\hat{x}(0) = \hat{y}(0) = 1$ and has solution

$$\hat{x}(t) = e^{-t}, \qquad \hat{y}(t) = e^{1-e^{-t}}.$$

However, $\hat{x}(t)$ and $\hat{y}(t)$ have nonzero initial values, and real-time CRN computable requires that all species to be initialized to 0. We rectify this by using Theorem 3.3.1. By introducing two new variables $x(t)$ and $y(t)$ satisfying $\hat{x} = x + 1$ and $\hat{y} = y + 1$, we obtain the PIVP

$$\frac{dx}{dt} = -(x+1), \qquad \frac{dy}{dt} = -(x+1)(y+1),$$

with $x(0) = y(0) = 0$ which meets this requirement.

To account for the negative terms in the ODE for $y(t)$, we apply the construction of Theorem 3.2.10 and introduce the variables u_x, v_x, u_y, v_y, carefully ensuring that $x = u_x - v_x$ and $y = u_y - v_y$. We can then write

$$\frac{dx}{dt} = -(x+1) = v_x - (u_x + 1) = p_x^+ - p_x^-,$$

where $p_x^+ = v_x$ and $p_x^- = u_x + 1$ are the positive and negative terms of $\frac{dx}{dt}$, respectively. Similarly, we can write

$$\begin{aligned}\frac{dy}{dt} &= -(x+1)(y+1) = -(u_x - v_x + 1)(u_y - v_y + 1) \\ &= -[(1+u_x)(1+u_y) + v_x v_y] + [(1+u_x)v_y + (1+u_y)v_x] \\ &= p_y^+ - p_y^-,\end{aligned}$$

where $p_y^- = (1+u_x)(1+u_y) + v_x v_y$ and $p_y^+ = (1+u_x)v_y + (1+u_y)v_x$ are the positive and negative terms of $\frac{dy}{dt}$, respectively.

We now define the ODEs for the variables u_x, v_x, u_y, v_y to be the following.

$$u_x' = p_x^+ - u_x v_x (p_x^+ + p_x^-), \tag{3.4.1}$$

$$v_x' = p_x^- - u_x v_x (p_x^+ + p_x^-), \tag{3.4.2}$$

$$u_y' = p_y^+ - u_y v_y (p_y^+ + p_y^-), \tag{3.4.3}$$

$$v_y' = p_y^- - u_y v_y (p_y^+ + p_y^-), \tag{3.4.4}$$

We note that $\frac{dx}{dt} = \frac{du_x}{dt} - \frac{dv_x}{dt}$ and $\frac{dy}{dt} = \frac{du_y}{dt} - \frac{dv_y}{dt}$, therefore if all of the variables are initialized to zero, we maintain our property that $x = u_x - v_x$ and $y = u_y - v_y$. The above PIVP is implementable by a CRN and we know that $u_y - v_y$ will exponentially converge to $e - 1$.

The final step of the construction is to apply the construction for closure under subtraction and addition to obtain a species that computes e directly. Thus, we introduce a species variable $z(t)$ and $z_r(t)$ with ODEs

$$z_r' = 1 - (u_y - v_y + 1) z_r, \tag{3.4.5}$$

$$z' = 1 - z z_r \tag{3.4.6}$$

We can now convert these ODEs to the CRN that implements them. Below, are all the species and reactions that induce the system of ODEs described above.

Reactions resulting from p_x^+ term in Equation (3.4.1) and p_x^- term in Equation (3.4.2):

$$V_x \xrightarrow{1} V_x + U_x, \qquad \emptyset \xrightarrow{1} V_x,$$
$$U_x \xrightarrow{1} U_x + V_x.$$

Reactions resulting from $-u_x v_x(p_x^+ + p_x^-)$ term in Equation (3.4.1) and p_x^- term in Equation (3.4.2):

$$U_x + V_x \xrightarrow{1} \emptyset, \qquad 2U_x + V_x \xrightarrow{1} U_x$$
$$U_x + 2V_x \xrightarrow{1} V_x.$$

Reactions resulting from p_y^+ term in Equation (3.4.3):

$$\emptyset \xrightarrow{1} U_y, \qquad U_x \xrightarrow{1} U_x + U_y,$$
$$U_y \xrightarrow{1} 2U_y, \qquad U_x + U_y \xrightarrow{1} U_x + 2U_y,$$
$$V_x + V_y \xrightarrow{1} V_x + V_y + U_y.$$

Reactions resulting from p_y^- term in Equation (3.4.4):

$$V_x \xrightarrow{1} V_x + V_y, \qquad U_x + U_y \xrightarrow{1} U_x + U_y + V_y,$$
$$V_y \xrightarrow{1} 2V_y, \qquad U_x + V_y \xrightarrow{1} U_x + 2V_y.$$

Reactions resulting from $-u_y v_y(p_y^+ + p_y^-)$ term in Equation (3.4.3) and (3.4.4):

$$U_y + V_y \xrightarrow{1} \emptyset, \qquad U_x + U_y + V_y \xrightarrow{1} U_x,$$
$$2U_y + V_y \xrightarrow{1} U_y, \qquad U_x + 2U_y + V_y \xrightarrow{1} U_x + U_y,$$
$$V_x + U_y + 2V_y \xrightarrow{1} V_x + V_y, \qquad V_x + U_y + V_y \xrightarrow{1} V_x,$$
$$V_x + 2U_y + V_y \xrightarrow{1} V_x + U_y, \qquad U_y + 2V_y \xrightarrow{1} V_y,$$
$$U_x + U_y + 2V_y \xrightarrow{1} U_x + V_y.$$

Reactions resulting from Equation (3.4.5) and (3.4.6).

$$\emptyset \xrightarrow{1} Z_r, \qquad Z_r + U_y \xrightarrow{1} U_y$$
$$Z_r + V_y \xrightarrow{1} 2Z_r + V_y, \qquad Z_r \xrightarrow{1} \emptyset$$
$$\emptyset \xrightarrow{1} Z, \qquad Z + Z_r \xrightarrow{1} Z_r.$$

To demonstrate that the above CRN actually computes e, we used *MATLAB* along with the *Simbiology* package to simulate it which is visualized in Figure 3.2.

It is easy to see that the species Z is indeed converging to e exponentially quickly.

3.4.2 Construction and Simulation of π

In our proof of Theorem 3.3.3, we used a CRN with integral initial conditions. In this section we construct the CRN with all zero initial conditions. Recall that the PIVP induced by our CRN used in Theorem 3.3.3 is

$$\hat{w}' = -\hat{w}, \tag{3.4.7}$$

$$\hat{x}' = -2\hat{w}\hat{x}\hat{y}, \tag{3.4.8}$$

$$y' = \hat{w}\hat{x}^2 - \hat{w}y^2, \tag{3.4.9}$$

$$z' = \hat{w}\hat{x}, \tag{3.4.10}$$

with $\hat{w}(0) = \hat{x}(0) = 1$ and $y(0) = z(0) = 0$.

According to our construction used in Theorem 3.3.1, we introduce variables w, x such that $\hat{w} = w + 1$ and $\hat{x} = x + 1$ to obtain the system of ODEs

$$w' = -(w + 1), \tag{3.4.11}$$

$$x' = -2(w + 1)(x + 1)y, \tag{3.4.12}$$

$$y' = (w + 1)(x + 1)^2 - (w + 1)y^2, \tag{3.4.13}$$

$$z' = (w + 1)(x + 1), \tag{3.4.14}$$

with initial condition $w(0) = x(0) = y(0) = z(0) = 0$.

Since many of the ODEs have negative terms preventing it from being implemented directly by a CRN, we follow the construction from Theorem 3.2.10 and introduce variables u_i, v_i for $i \in \{1, 2, 3, 4\}$ that satisfy

$$w = u_1 - v_1, \qquad x = u_2 - v_2, \qquad y = u_3 - v_3, \qquad z = u_4 - v_4.$$

We now substitute the variables u_i, v_i into equations (3.4.11)-(3.4.14) and separate the positive and negative terms.

Performing the substitution to equation (3.4.11), we obtain

$$w' = v_1 - (1 + u_1) = p_1^+ - p_1^-,$$

where $p_1^+ = v_1$ and $p_1^- = 1 + u_1$ are the positive and negative terms of w', respectively. Performing the substitution into (3.4.12) and separating positive and negative terms yields

$$\begin{aligned} p_2^+ &= 2v_3(u_1+1)(u_2+1) + 2v_1v_2v_3 + 2(u_1+1)v_2u_3 + 2v_1(u_2+1)u_3, \\ p_2^- &= 2(u_1+1)v_2v_3 + 2v_1(u_2+1)v_3 + 2(u_1+1)(u_2+1)u_3 + 2v_1v_2u_3. \end{aligned}$$

Performing the substitution into (3.4.13) and separating positive and negative terms yields

$$\begin{aligned} p_3^+ &= u_1(u_2+1)^2 + u_1v_2^2 + 2v_1v_2(u_2+1) + v_1u_3^2 + v_1v_3^2 \\ &\quad + 2u_1u_3v_3 + (u_2+1)^2 + v_2^2 + 2u_3v_3 \\ p_3^- &= v_1(u_2+1)^2 + v_1v_2^2 + 2u_1(u_2+1)v_2 + u_1u_3^2 + u_1v_3^2 \\ &\quad + 2v_1u_3v_3 + 2(1+u_2)v_2 + u_3^2 + v_3^2 \end{aligned}$$

Finally, performing the substitution into (3.4.14) and separating positive and negative terms yields Again, for terms resulting from (3.4.14), we let

$$\begin{aligned} p_4^+ &= (u_1+1)(u_2+1) + v_1v_2, \\ p_4^- &= v_1(1+u_2) + v_2(1+u_1). \end{aligned}$$

Then the following ODE system can be implemented by a CRN.

$$u_1' = p_x^+ - u_1 v_1 (p_x^+ + p_x^-), \tag{3.4.15}$$

$$v_1' = p_x^- - u_1 v_1 (p_x^+ + p_x^-), \tag{3.4.16}$$

$$u_2' = p_2^+ - u_2 v_2 (p_2^+ + p_2^-), \tag{3.4.17}$$

$$v_2' = p_2^- - u_2 v_2 (p_2^+ + p_2^-), \tag{3.4.18}$$

$$u_3' = p_3^+ - u_3 v_3 (p_3^+ + p_3^-), \tag{3.4.19}$$

$$v_3' = p_3^- - u_3 v_3 (p_3^+ + p_3^-), \tag{3.4.20}$$

$$u_4' = p_4^+ - u_4 v_4 (p_4^+ + p_4^-), \tag{3.4.21}$$

$$v_4' = p_4^- - u_4 v_4 (p_4^+ + p_4^-), \tag{3.4.22}$$

where all variables are initialized to be zero.

We have already shown that $z(t) = \arctan(1 - e^{-t}) = u_4(t) - v_4(t)$ which converges to $\frac{\pi}{4}$ in real-time. Thus, our last step is to apply the construction for subtraction by adding the variables p_r and p with ODEs:

$$p_r' = 1 - 4(u_4 - v_4) p_r, \tag{3.4.23}$$

$$p' = 1 - p p_r \tag{3.4.24}$$

Then by Lemma 3.2.3, we know that p_r' converges to $\frac{1}{\pi}$ in real-time, and p converges to π in real-time.

The complete list of reactions that implement (3.4.15)-(3.4.23) are listed as below. Reactions resulting from p_x^+ and p_2^-:

$$V_1 \xrightarrow{1} V_1 + U_1, \qquad \emptyset \xrightarrow{1} V_1,$$

$$U_1 \xrightarrow{1} U_1 + V_1.$$

Reactions resulting from $-u_1 v_1 (p_x^+ + p_x^-)$:

$$U_1 + 2V_1 \xrightarrow{1} V_1, \qquad U_1 + V_1 \xrightarrow{1} \emptyset$$

$$2U_1 + V_1 \xrightarrow{1} U_1.$$

Reactions resulting from p_2^+:

$$V_3 \xrightarrow{2} V_3 + U_2, \qquad V_3 + U_1 \xrightarrow{2} V_3 + U_1 + U_2,$$
$$V_3 + U_2 \xrightarrow{2} V_3 + 2U_2, \qquad V_3 + U_1 + U_2 \xrightarrow{2} V_3 + U_1 + 2U_2,$$
$$V_1 + V_2 + V_3 \xrightarrow{2} V_1 + V_2 + V_3 + U_2, \qquad U_1 + V_2 + U_3 \xrightarrow{2} U_1 + U_2 + V_2 + U_3,$$
$$V_2 + U_3 \xrightarrow{2} U_2 + V_2 + U_3, \qquad V_1 + U_3 \xrightarrow{2} V_1 + U_2 + U_3,$$
$$V_1 + U_2 + U_3 \xrightarrow{2} V_1 + 2U_2 + U_3.$$

Reactions resulting from p_2^-:

$$U_1 + V_2 + V_3 \xrightarrow{2} U_1 + 2V_2 + V_3, \qquad V_2 + V_3 \xrightarrow{2} 2V_2 + V_3,$$
$$V_1 + V_3 \xrightarrow{2} V_1 + V_2 + V_3, \qquad V_1 + U_2 + V_3 \xrightarrow{2} V_1 + U_2 + V_2 + V_3,$$
$$U_1 + U_2 + U_3 \xrightarrow{2} U_1 + U_2 + V_2 + U_3, \qquad U_1 + U_3 \xrightarrow{2} U_1 + V_2 + U_3,$$
$$U_2 + U_3 \xrightarrow{2} U_2 + V_2 + U_3, \qquad U_3 \xrightarrow{2} V_2 + U_3,$$
$$V_1 + V_2 + U_3 \xrightarrow{2} V_1 + 2V_2 + U_3.$$

Reactions resulting from $-u_2 v_2 p_2^+$:

$$U_2 + V_2 + V_3 \xrightarrow{2} V_3, \qquad U_2 + V_2 + V_3 + U_1 \xrightarrow{2} V_3 + U_1,$$
$$2U_2 + V_2 + V_3 \xrightarrow{2} U_2 + V_3, \qquad U_1 + 2U_2 + V_2 + V_3 \xrightarrow{2} U_1 + U_2 + V_3,$$
$$V_1 + U_2 + 2V_2 + V_3 \xrightarrow{2} V_1 + V_2 + V_3, \qquad U_1 + U_2 + 2V_2 + U_3 \xrightarrow{2} U_1 + V_2 + U_3,$$
$$U_2 + 2V_2 + U_3 \xrightarrow{2} V_2 + U_3, \qquad V_1 + U_2 + V_2 + U_3 \xrightarrow{2} V_1 + U_3,$$
$$V_1 + 2U_2 + V_2 + U_3 \xrightarrow{2} V_1 + U_2 + U_3.$$

Reactions resulting from $-u_2 v_2 p_2^-$:

$$U_1 + U_2 + 2V_2 + V_3 \xrightarrow{2} U_1 + V_2 + V_3, \qquad U_2 + 2V_2 + V_3 \xrightarrow{2} V_2 + V_3,$$
$$V_1 + U_2 + V_2 + V_3 \xrightarrow{2} V_1 + V_3, \qquad V_1 + 2U_2 + V_2 + V_3 \xrightarrow{2} V_1 + U_2 + V_3,$$
$$U_1 + 2U_2 + V_2 + U_3 \xrightarrow{2} U_1 + U_2 + U_3, \qquad U_1 + U_2 + V_2 + U_3 \xrightarrow{2} U_1 + U_3,$$
$$2U_2 + V_2 + U_3 \xrightarrow{2} U_2 + U_3, \qquad U_2 + V_2 + U_3 \xrightarrow{2} U_3,$$
$$V_1 + U_2 + 2V_2 + U_3 \xrightarrow{2} V_1 + V_2 + U_3.$$

Reactions resulting from p_3^+:

$$U_1 \xrightarrow{1} U_1 + U_3, \qquad U_1 + U_2 \xrightarrow{2} U_1 + U_2 + U_3,$$
$$U_1 + 2U_2 \xrightarrow{1} U_1 + 2U_2 + U_3, \qquad U_1 + 2V_2 \xrightarrow{1} U_1 + 2V_2 + U_3,$$
$$V_1 + V_2 \xrightarrow{2} V_1 + V_2 + U_3, \qquad V_1 + V_2 + U_2 \xrightarrow{2} V_1 + U_2 + V_2 + U_3,$$
$$V_1 + 2U_3 \xrightarrow{1} V_1 + 3U_3, \qquad V_1 + 2V_3 \xrightarrow{1} V_1 + 2V_3 + U_3,$$
$$U_1 + U_3 + V_3 \xrightarrow{2} U_1 + 2U_3 + V_3, \qquad 2U_2 \xrightarrow{1} 2U_2 + U_3,$$
$$U_2 \xrightarrow{2} U_2 + U_3, \qquad \emptyset \xrightarrow{1} U_3,$$
$$2V_2 \xrightarrow{1} 2V_2 + U_3, \qquad U_3 + V_3 \xrightarrow{2} 2U_3 + V_3.$$

Reactions resulting from p_3^-:

$$V_1 \xrightarrow{1} V_1 + V_3, \qquad V_1 + U_2 \xrightarrow{2} V_1 + U_2 + V_3,$$
$$V_1 + 2U_2 \xrightarrow{1} V_1 + 2U_2 + V_3, \qquad V_1 + 2V_2 \xrightarrow{1} V_1 + 2V_2 + V_3,$$
$$U_1 + V_2 \xrightarrow{2} U_1 + V_2 + V_3, \qquad U_1 + U_2 + V_2 \xrightarrow{2} U_1 + U_2 + V_2 + V_3,$$
$$U_1 + 2U_3 \xrightarrow{1} U_1 + 2U_3 + V_3, \qquad U_1 + 2V_3 \xrightarrow{1} U_1 + 3V_3,$$
$$V_1 + U_3 + V_3 \xrightarrow{2} V_1 + U_3 + 2V_3, \qquad V_2 \xrightarrow{2} V_2 + V_3,$$
$$V_2 + U_2 \xrightarrow{2} V_2 + U_2 + V_3, \qquad 2U_3 \xrightarrow{1} 2U_3 + V_3,$$
$$2V_3 \xrightarrow{1} 3V_3$$

Reactions resulting from $-u_3v_3p_3^+$:

$$U_1+U_3+V_3 \xrightarrow{1} U_1, \qquad U_1+U_2+U_3+V_3 \xrightarrow{2} U_1+U_2,$$
$$U_1+2U_2+U_3+V_3 \xrightarrow{1} U_1+2U_2, \qquad U_1+2V_2+U_3+V_3 \xrightarrow{1} U_1+2V_2,$$
$$V_1+V_2+U_3+V_3 \xrightarrow{2} V_1+V_2, \qquad V_1+U_2+V_2+U_3+V_3 \xrightarrow{2} V_1+U_2+V_2,$$
$$V_1+3U_3+V_3 \xrightarrow{1} V_1+2U_3, \qquad V_1+U_3+3V_3 \xrightarrow{1} V_1+2V_3,$$
$$U_1+2U_3+2V_3 \xrightarrow{2} U_1+U_3+V_3, \qquad 2U_2+U_3+V_3 \xrightarrow{1} 2U_2,$$
$$U_2+U_3+V_3 \xrightarrow{2} U_2, \qquad U_3+V_3 \xrightarrow{1} \emptyset,$$
$$2V_2+U_3+V_3 \xrightarrow{1} 2V_2, \qquad 2U_3+2V_3 \xrightarrow{2} U_3+V_3.$$

Reactions resulting from $-u_3v_3p_3^-$:

$$V_1+U_3+V_3 \xrightarrow{1} V_1, \qquad V_1+U_2+U_3+V_3 \xrightarrow{2} U_1+V_1,$$
$$V_1+2U_2+U_3+V_3 \xrightarrow{1} V_1+2U_2, \qquad V_1+2V_2+U_3+V_3 \xrightarrow{1} V_1+2V_2,$$
$$U_1+V_2+U_3+V_3 \xrightarrow{2} U_1+V_2, \qquad U_1+U_2+V_2+U_3+V_3 \xrightarrow{2} U_1+U_2+V_2,$$
$$U_1+3U_3+V_3 \xrightarrow{1} U_1+2U_3, \qquad U_1+U_3+3V_3 \xrightarrow{1} U_1+2V_3,$$
$$V_1+2U_3+2V_3 \xrightarrow{2} V_1+U_3+V_3, \qquad V_2+U_3+V_3 \xrightarrow{2} V_2,$$
$$U_2+V_2+U_3+V_3 \xrightarrow{2} U_2+V_2, \qquad 3U_3+V_3 \xrightarrow{1} 2U_3,$$
$$U_3+3V_3 \xrightarrow{1} 2V_3.$$

Reactions resulting from p_4^+:

$$\emptyset \xrightarrow{1} U_4, \qquad U_1 \xrightarrow{1} U_1+U_4,$$
$$U_2 \xrightarrow{1} U_2+U_4, \qquad U_1+U_2 \xrightarrow{1} U_1+U_2+U_4,$$
$$V_1+V_2 \xrightarrow{1} V_1+V_2+U_4.$$

Reactions resulting from p_4^-:

$$V_1 \xrightarrow{1} V_1+V_4, \qquad V_1+U_2 \xrightarrow{1} V_1+U_2+V_4,$$
$$V_2 \xrightarrow{1} V_2+V_4, \qquad U_1+V_2 \xrightarrow{1} U_1+V_2+V_4.$$

Reactions resulting from $-u_4 v_4(p_4^+ + p_4^-)$:

$$U_4 + V_4 \xrightarrow{1} \emptyset, \qquad U_1 + U_4 + V_4 \xrightarrow{1} U_1,$$
$$U_2 + U_4 + V_4 \xrightarrow{1} U_2, \qquad U_1 + U_2 + U_4 + V_4 \xrightarrow{1} U_1 + U_2,$$
$$V_1 + V_2 + U_4 + V_4 \xrightarrow{1} V_1 + V_2, \qquad V_1 + U_4 + V_4 \xrightarrow{1} V_1,$$
$$V_1 + U_2 + U_4 + V_4 \xrightarrow{1} V_1 + U_2, \qquad V_2 + U_4 + V_4 \xrightarrow{1} V_2,$$
$$U_1 + V_2 + U_4 + V_4 \xrightarrow{1} U_1 + V_2.$$

Reactions resulting from (3.4.23) and (3.4.24):

$$\emptyset \xrightarrow{1} P_r, \qquad P_r + U_4 \xrightarrow{4} U_4,$$
$$P_r + V_4 \xrightarrow{4} 2P_r + V_4, \qquad \emptyset \xrightarrow{1} P$$
$$P + P_r \xrightarrow{1} P_r$$

Figure 3.3 shows a MATLAB/Simbiology simulation of the above reactions and visually demonstrates that the species P does in fact converge to π exponentially quickly.

3.5 Discussion

We showed the real-time equivalence of CRNs and analog computers and use the construction to give natural proof of transcendental numbers like e and π being CRN-computable in real time. The essential trick of showing the later is to take some analytic functions, e.g., $f(t) = e^t$ and $g(t) = \arctan(t)$, and then take special values: $f(1) = e$ and $g(1) = \frac{\pi}{4}$. If we can converge to the special values fast, then we can converge to the function value fast. That is why we pick $1 - e^{-t}$ to converge to one. And the function in consideration become $f_1(t) = e^{1-e^{-t}}$ and $g_1(t) = \arctan(1 - e^{-t})$. The rest of the work is just to code $f_1(t)$ and $g_1(t)$ into PIVPs.

All the functions that can be generated by CRNs or GPACs are analytic functions, we call this collection of function $\mathcal{F}$ and use $\mathcal{F}(\mathbb{Z})$ to denote the functions value at integers. By our definition of $\mathbb{R}_{RTCRN}$ and the above argument we can see $\mathbb{R}_{RTCRN} = \mathcal{F}(\mathbb{Z})$. What if we want to use numbers in

$\mathcal{F}(\mathbb{Z})$ as initial value? Can we compute any new numbers? That is

$$\mathbb{R}_{RTCRN} \stackrel{?}{=} \bigcup_i \mathcal{F}^i(\mathbb{Z}).$$

In [45], Pouly called the right-hand size set *generable numbers* by GPACs. We tend to believe the two sets equal, but it seems to require new insights to prove this fact and it will be left to a motivated future researcher.

CHAPTER 4. POPULATION PROTOCOL COMPUTABLE NUMBERS

In this chapter we continue the discussion of computability of real numbers but we restrict our discussion to a special model of CRNs called Population Protocols (PPs). We revisit the notion of Large-Population Protocols (LPPs) computable numbers from [3, 30]. We modify the notion to capture the computability of transcendental numbers by LPPs.

4.1 LPP-Computable Number Revisit

A Population Protocol can be viewed as a special CRN where the reactions must have exactly two reactants and two products. That is, the reactions must have the form

$$q_i + q_j \longrightarrow q_k + q_l.$$

Sometimes we also call the above a transition rule and the above transition will be denoted as $(q_k, q_l) = \Delta(q_i, q_j)$, where Δ is the transition function.

What are called Large-Population Protocols in [3] can be thought of as a different analytical setting rather than a new computational model. That is, LPPs are essentially PPs by definition, but they deal with populations which grow to infinity.

There are several other new mechanisms and restrictions in the definition of LPP-computable number in [3].

1. The quantity they use to compute a number is not the population of a species and we know the population can only be positive integers. Instead, they use the *fraction* of the species in the total population. An obvious and yet important fact is that the sum of these fractions is one. This fact is critical later in the chapter.

   Note that for every fixed population the fractions must be rationals. However, when the population varies and goes to infinity, the limits of these fractions can be non-rationals. The

fact that the quantities are fractions of the population make it unviable to perform arithmetic operations like addition and multiplication as in the previous chapters. For example, say species A computes $\frac{1}{2}$ and species B computes $\frac{2}{3}$. There is no way one can construct a species C to compute $\frac{1}{2} + \frac{2}{3}$, which is greater than one. Also, adding a new species C will change the fraction of A and B in the population.

2. An alternative way to do addition is to allow a user to specify a set of marked states (species, variables), the sum of which will be used to compute the target number.

3. Another restriction in their definition is to require that the ODE systems associated with the PPs must have *finitely* many fixed points. This restriction is useful in stability analysis and is critical in characterizing the LPP-computable numbers. However, many useful ODE systems do not have finitely many fixed points and hence are ruled out.

The formal definition of an LPP computable number is given below. Note that the terminologies are slightly different from the CRN setting. A reader familiar with CRNs can identify the set of states, Q, to be the counterpart of the set of species or variables in a CRN. We must also distinguish (q_i, q_j) and (q_j, q_i) when they are on the left-hand side of the reaction; that is, they do not necessarily end up in the same products.

Definition 1. *[3] A real number ν is said to be computable by an LPP if there exists a vector $x^* = (x_1, x_2, \cdots, x_k) \in [0,1]^k$ such that $\sum_i^k x_i = 1$, and an LPP $\mathcal{P}$, admitting finitely many equilibria, such that $(x_1, x_2, \cdots, x_k)$ is a stable equilibrium of $\mathcal{P}$ and $\sum_{q_i \in Q^+} x_i = \nu$, where Q^+ is the set of marked states for $\mathcal{P}$.*

A note originally in [3] discusses the reason for the requirement of finitely many equilibria. This assumption is needed to *"avoid pathological cases, in particular the case of idle systems $q\ q' \to q\ q'$ for all q and q'. Indeed, in idle systems, all initial states are equilibria, and such a system would compute any real of [0,1], depending on the initial configuration."*

While this extreme case does show something pathological, it is a bit of an overkill since one can prevent this from happening by restricting the initial configuration to be rational numbers, just like what we did in Chapter 2 and 3 of this thesis.

LPP-computable numbers defined as above are exactly the algebraic numbers.

Theorem 4.1.1. *[3] Every $\nu \in [0,1]$ is computable by an LPP if and only if it is algebraic.*

We relax the definition of an LPP computable number by removing the finitary requirement about fixed points. From this point on, we adopt the following definition instead.

Definition 2. *A real number ν is said to be computable by an LPP if there exists a vector $x^* = (x_1, x_2, \cdots, x_k) \in [0,1]^k$ such that $\sum_i^k x_i = 1$ and $\sum_{q_i \in Q^+} x_i = \nu$, where Q^+ is the set of marked states for $\mathcal{P}$. All the states x_i must be initialized to some positive rational $r_i \in \mathbb{Q} \cap [0,1]$, in the sense that $\lim_{n\to\infty} x_i^{(n)}(0) = r_i$, when $x_i^{(n)}(0)$ is the initial fraction of state i at the stage that the population is n.*

Next, we will construct a LPP that computes $\frac{\pi}{4}$. (LPP-computable numbers have to be smaller than 1.)

4.2 LPP Computability Under The New Definition: the Computation of $\frac{\pi}{4}$

We break the construction down into several steps.

1. Select a formula of $\frac{\pi}{4}$.
2. Encode the formula in to a PIVP.
3. Rewrite the PIVP in the previous step into a degree-two polynomial.
4. Rewrite the last system again in to a degree-two homogeneous polynomial.
5. Turn this system into a PP.

We will discuss the steps in detail in the rest of the chapter.

4.2.1 Step 1: Machin-Like Formulas

We use the following so-called Machin-like formulas

$$\frac{\pi}{4} = 2\arctan(1/3) + \arctan(1/7) \tag{4.2.1}$$

to start our construction. A curious reader can learn more about this type of formula at this site [1]. Note that we cannot use the one we adopted in Chapter 3, $\frac{\pi}{4} = \arctan(1 - e^{-t})$, or basically $\frac{\pi}{4} = \arctan(1)$, since the setting of LPP-computable numbers requires the sum of all the variables to be 1. If we use that formula, we must use at least one variable for $1 - e^{-t}$, which will be very close to one, and another variable for the final result $\frac{\pi}{4}$. The sum of the variables will already be more than one.

To make it clear, we make the observation below.

Observation 4.2.1. *In an LPP the sum of all variables must be one at all times.*

4.2.2 Step 2: Construct a PIVP

We've seen a PIVP that computes $\arctan(x)$ before. Here we use a similar system.

We first show how to compute $2\arctan\left(\frac{1}{3}\right)$.

$$\begin{cases} X' = -2X, \\ Z' = X, \\ V' = (1-V)^2 \cdot 2Z \cdot X, \\ I' = 2VX, \\ W' = -\sum_{x \neq W} x', \quad x \text{ ranges over all variables except } W \text{ itself.} \end{cases} \tag{4.2.2}$$

We first let $X(0) = \frac{2}{3}$, $I(0) = V(0) = Z(0) = W(0) = 0$.

Note that the way we construct W' guarantees the derivatives of all variables sum to zero. So the total mass of the system does not change.

One can verify that

$$\begin{cases} X = \frac{2}{3}e^{-2t}, \\ Z = \frac{1}{3}(1 - e^{-2t}), \\ V = \frac{Z^2}{1+Z^2}, \\ I = \int I' \,\mathrm{d}t = \int 2VX \,\mathrm{d}t = 2\int \frac{Z^2}{1+Z^2} \,\mathrm{d}Z, \\ W' = -\sum_{x \neq W} x_i', \quad x \text{ ranges over all variables except } W \text{ itself.} \end{cases} \tag{4.2.3}$$

Note that in the limit as t goes to infinity, we have

$$\begin{cases} X_\infty = 0, \\ Z_\infty = 1/3, \\ V_\infty = \frac{Z_\infty^2}{1+Z_\infty^2} = 1/10, \\ I_\infty = 2\int_0^{1/3} \frac{Z^2}{1+Z^2} \,\mathrm{d}Z = \frac{2}{3} - 2\arctan\left(\frac{1}{3}\right) \\ W_\infty = \frac{2}{3} - X_\infty - Z_\infty - V_\infty - I_\infty. \end{cases} \tag{4.2.4}$$

Next, if we mark $\{Z, V, W\}$, and since we know the total sum of all variables is always $\frac{2}{3}$, we have

$$Z_\infty + V_\infty + W_\infty = \frac{2}{3} - (\frac{2}{3} - 2\arctan\left(\frac{1}{3}\right)) = 2\arctan\left(\frac{1}{3}\right).$$

We will compute the other part of 4.2.1, $\arctan\left(\frac{1}{7}\right)$, in a similar fashion. We have

$$\begin{cases} \bar{X}' = -2\bar{X}, \\ \bar{Z}' = \bar{X}, \\ \bar{V}' = (1 - \bar{V})^2 \cdot 2\bar{Z} \cdot \bar{X}, \\ \bar{I}' = \bar{V}\bar{X}, \quad \text{note that we don't have the factor of 2 here;} \\ \bar{W}' = -\sum_{x_i \neq \bar{W}} x_i', \quad x_i \text{ ranges over all variables except } \bar{W} \text{ itself.} \end{cases} \tag{4.2.5}$$

and let $\bar{X}(0) = \frac{2}{7}$, $\bar{I}(0) = \bar{V}(0) = \bar{Z}(0) = \bar{W}(0) = 0$. The solution is similar to the system 4.2.2. In the limit, we have

$$\begin{cases} \bar{X}_\infty = 0, \\ \bar{Z}_\infty = 1/7, \\ \bar{V}_\infty = 1/50, \\ \bar{I}_\infty = \frac{1}{7} - \arctan\left(\frac{1}{7}\right) \\ \bar{W}_\infty = \frac{2}{7} - X_\infty - Z_\infty - V_\infty - I_\infty. \end{cases} \tag{4.2.6}$$

and we mark $\{\bar{V}, \bar{W}\}$. It is easy to see that

$$V_\infty + W_\infty = \arctan\left(\frac{1}{7}\right).$$

We combine System 4.2.2 and System 4.2.5 together and mark $\{Z, V, W, \bar{V}, \bar{W}\}$. We then get a system with marked variables that computes $\frac{\pi}{4}$.

It is crucial that $\frac{2}{3} + \frac{2}{7} = \frac{20}{21} < 1$. For the rest of the "mass", i.e., the quantity $1 - \frac{20}{21} = \frac{1}{21}$, we can introduce a idle variable that does not interact with other variable and we initialize it to $\frac{1}{21}$. In this way we maintain the sum of all variables to be one.

For the remaining steps, we will just use System 4.2.2 to illustrate the idea, since the processes for the two systems 4.2.2 and 4.2.5 are about the same.

4.2.3 Step 3: Rewrite the PVIP into a Degree-Two Polynomial System

The PVIP in System 4.2.2 can not be implemented directly by a PP since some terms consist of more than two variables. The first thing we need to do to fix this is to rewrite the terms in the polynomial such that no term consists of more than two variables. The idea is a straightforward one: For a term XYZ, we can introduce a new variable W such that $W = YZ$ and rewrite the term XYZ to XW. The process can generally be done by Theorem 4 of [13]. However, the construction introduces too many new variables and can not be applied here. For a term or monomial with the

form

$$y_1^{i_1} y_2^{i_2} \cdots y_n^{i_n},$$

one has to introduce $i_1 \cdot i_2 \cdots i_n$ many variables. Some of these variables have non-zero initial values and will take some share of the total mass of one in the LPP model. Therefore we have to plan carefully what new variables to introduce while rewriting the system.

We introduce new variables K, R, J, and P in System 4.2.2 such that

$$K = ZV, \quad R = XZ, \quad J = XV, \quad \text{and } P = XK.$$

Take the derivatives of K, R, J, and P and add them into System 4.2.2. We have

$$\begin{cases} X' = -2X, \\ Z' = X, \\ V' = 2XZ - 4VR + 2JK, \\ I' = 2VX, \\ K' = J + 2ZR - 4KR + 2PK, \\ J' = -2J + 2XR - 4JK + 2JP, \\ R' = -2R + X^2, \\ W' = -\sum_{x \neq W} x', \quad x \text{ range over all variables except } W \text{ itself.} \end{cases} \tag{4.2.7}$$

4.2.4 Step 4: Rewrite the PVIP into a Degree-Two Homogeneous Polynomial System

We want to do this because all "reactions" in PPs have exactly two reactants. System 4.2.7 still contains some degree-one terms. Some other system may contain constant terms at this stage. We need to rewrite such terms.

Observation 4.2.1 can help to achieve that.

Suppose we have a variable set $x_1, x_2, \ldots, x_n$.

1. To rewrite a degree-one term x_i, we just need to write

$$x_i = x_i \cdot 1 = x_i \cdot (x_1 + x_2 + \ldots + x_n).$$

2. To rewrite a constant term a, we can write

$$a = a \cdot 1 \cdot 1 = a(x_1 + x_2 + \ldots + x_n)(x_1 + x_2 + \ldots + x_n).$$

One can apply the above two rules to rewrite System 4.2.7. For the sake of simplicity, we do not list the new system here and will refer to this system as System $\mathcal{P}$. Later in this chapter we will need a parametrized version of it relative to a parameter ε. We call such a system $\mathcal{P}_\varepsilon$.

4.2.5 Step 5: Translate the Degree-Two Homogeneous PIVP into an LPP

Finally, we need to turn the PIVP we get from the previous step into a PP. We need to take a detour: We will first turn the PIVP into a probabilistic LPP (PLPP) then derandomize the PLPP into an LPP. The later can be done by the construction in Lemma 15 of [30]. So we only need to focus on the former process.

We first revisit some useful concepts regarding PPs. Let Q be a set of states in a PP. The *balance* equation defined in [3] is a function $b : \mathbb{R}^{|Q|} \to \mathbb{R}^{|Q|}$ such that

$$b(x) = \sum_{(q_1,q_2)\in Q^2} \left(x_{q_1} x_{q_2} \big(- e_{q_1} - e_{q_2} + \sum_{(q_3,q_4)\in Q^2} \delta_{q_1,q_2,q_3,q_4} (e_{q_3} + e_{q_4}) \big) \right) \tag{4.2.8}$$

where $\delta_{q_1,q_2,q_3,q_4} = 1$ if $\Delta(q_1, q_2) = (q_3, q_4)$, and 0 otherwise; and $(e_q)_{q\in Q}$ is the canonical base of $\mathbb{R}^{|Q|}$. The balance equation is a description of the dynamics of the system. Intuitively, it says whenever q_1 and q_2 bump into each other, if there is a reaction $\Delta(q_1, q_2) = (q_3, q_4)$ in the PP, then the pair (q_1, q_2) turns into (q_3, q_4); otherwise, things do not change.

In the above, δ_{q_1,q_2,q_3,q_4} is either 0 or 1. Now we introduce probabilistic LPPs (PLPPs) and probabilistic transition rules. Basically, this means that s δ_{q_1,q_2,q_3,q_4} should be a probability distribution if we fix (q_1, q_2). Formally, for PLPPs, the transition rules have the form

$$q_i \; q_j \to \alpha_{i,j,k,l} \; q_k \; q_l$$

and for every $(q_i, q_j) \in Q^2$, we have

- for every $(q_k, q_l) \in Q^2$, $\alpha_{i,j,k,l} \in \mathbb{Q}$ and $\alpha_{i,j,k,l} > 0$, and
- $\sum_{(q_k,q_l)} \alpha_{i,j,k,l} = 1$.

For PLPPs, the balance function basically stay the same as 4.2.8.

$$b(x) = \sum_{(q_1,q_2)\in Q^2} \left(x_{q_1} x_{q_2} \big(- e_{q_1} - e_{q_2} + \sum_{(q_3,q_4)\in Q^2} \alpha_{q_1,q_2,q_3,q_4} (e_{q_3} + e_{q_4}) \big) \right) \tag{4.2.9}$$

One can find more details about this topic in [3].

The ODE associated with a PLPP can be written as

$$\frac{\mathrm{d}x}{\mathrm{d}t} = b(x),$$

where $x \in \mathbb{R}^Q$ and its q_i component, x_{q_i}, keeps track of the fraction of q_i in the population.

For the q component of the above equation, we have the following observation.

Observation 4.2.2. *The q component has the form*

$$\frac{\mathrm{d}x_q}{\mathrm{d}t} = f(x) - 2x_q, \tag{4.2.10}$$

where $f(x)$ is a degree-two homogeneous polynomial and every monomial has positive coefficient.

Proof. Take the x_q component in 4.2.9. We have

$$\begin{aligned}
\frac{\mathrm{d}x_q}{\mathrm{d}t} = & \sum_{(q_1=q,q_2)\in Q^2} \Big(- x_{q_1} x_{q_2} \Big) + \sum_{(q_1,q_2=q)\in Q^2} \Big(- x_{q_1} x_{q_2} \Big) \\
& + \sum_{(q_1,q_2)\in Q^2} \Big(x_{q_1} x_{q_2} \sum_{(q_3,q_4)\in Q^2} \alpha_{q_1,q_2,q_3,q_4} (e_{q_3} + e_{q_4}) \Big) \\
= & -2x_q + \sum_{(q_1,q_2)\in Q^2} \Big(x_{q_1} x_{q_2} \sum_{(q_3,q_4)\in Q^2} \alpha_{q_1,q_2,q_3,q_4} (e_{q_3} + e_{q_4}) \Big), \quad \text{by Observation 4.2.1.}
\end{aligned}$$

Let $f(x)$ be the x_q component of $\sum_{(q_1,q_2)\in Q^2} \Big(x_{q_1} x_{q_2} \sum_{(q_3,q_4)\in Q^2} \alpha_{q_1,q_2,q_3,q_4} (e_{q_3} + e_{q_4}) \Big)$. This completes the proof. □

An intuitive way to think about equation 4.2.10:

- The term $-2x_q$ represents all reactions that *consume* x_q. A complete set of rules of a PP must include all combinations of pairs (x_q, x_k), or (x_k, x_q), where $x_k \in Q$. These combinations contribute to the $-2x_q$ term.

- The $f(x)$ represents all the reactions that *produce* x_q. This observation is critical in our construction. We will see this soon.

With the above build-up of the necessary preliminaries, we are ready to continue the construction. After Step 4, we get a system $\mathcal{P}$. For simplicity of notation, we say $\mathcal{P}$ has the following form

$$x' = f(x), \tag{4.2.11}$$

where x is the variable vector (X, Z, V, I, K, J, R, W) for variables in System 4.2.5. However, for the sake of generality, we will view $x = (x_1, \cdots, x_n)$ as a set of variables $\{x_1, \cdots, x_n\}$ instead, and $f(x)$ as a degree-two homogeneous polynomial.

In order translate the PIVP into a PLPP, we need to bring down the coefficients to $[0, 1]$. Therefore, we use a parametrized version of system 4.2.11, $\mathcal{P}_\varepsilon$, such that

$$x' = \varepsilon f(x). \tag{4.2.12}$$

Note that this system only dilates time. The limiting behavior does not change.

For each component of Equation 4.2.12, we write

$$\begin{aligned} x_i' &= \varepsilon f_i(x) + 2x_i - 2x_i \\ &= \varepsilon f_i(x) + 2x_i(x_1 + \cdots + x_n) - 2x_i. \end{aligned} \tag{4.2.13}$$

Let

$$g_i(x) = \varepsilon f_i(x) + 2x_i(x_1 + \cdots + x_n). \tag{4.2.14}$$

We use the following steps to turn equation 4.2.12 into a PLPP. The main idea is a greedy one: If a term $x_k x_l$ appears in $g_i(x)$, this means the pair $x_k x_l$ will produce x_i. We use the all-in strategy to produce two x_i's.

Construction 4.2.1.

1. *Pick an ε such that after combining like monomials, the coefficient of every term with the form $x_i x_j$ in $g_i(x)$ becomes positive. This is always achievable since all negative terms must have an x_i factor for an ODE associated with a CRN or a PP. We just need to pick an ε small enough to ensure the $2x_i x_j$ terms dominate, for all x_j.*

2. *Consider the pair* (x_k, x_l). *For a term* $\alpha_{i,k,l}x_kx_l$ *in* $g_i(x)$ *we do the following:*

   - *If* $k = l$, *add a rule*

   $$x_k\ x_k \to \frac{\alpha_{i,k,l}}{2}x_i\ x_i;$$

   - *otherwise, add two rules*

   $$x_k\ x_l \to \frac{\alpha_{i,k,l}}{4}x_i\ x_i;$$

   and

   $$x_l\ x_k \to \frac{\alpha_{i,k,l}}{4}x_i\ x_i;$$

   We do this in the above way because, in the PP model used here, we distinguish the order of a pair except for pairs of the same variables.

It is easy to verify that the dynamics of the PLPP we constructed out of the above rules can be described by equation 4.2.13 and hence by equation 4.2.12.

There is still one fact that needs to be clarified:

Observation 4.2.3. *The protocol resuling from Construction 4.2.1 is a PLPP.*

Proof. Consider a term x_kx_l in $g_i(x)$ as defined in Equation 4.2.14. Let's first discuss the case $k \neq l$.

The term x_kx_l occurs in $g_i(x)$ from two sources

- Case 1: It comes from the $\varepsilon f_i(x)$ part. The sum of the coefficients of such terms for all i must be zero since this is the case for $\varepsilon f(x)$. Recall that we construct it to be this way.

- Case 2: It come from the $2x_i(x_1 + \cdots + x_n)$ part. We have two symmetric sources:

  $$2x_k(x_1 + \cdots x_l + \cdots + x_n)$$

  in $g_k(x)$, and

  $$2x_l(x_1 + \cdots x_k + \cdots + x_n)$$

  in $g_l(x)$. The sum of these two occurrences is $4x_kx_l$.

In the construction above, when we assign probability for a rule, we always divide coefficients by 4 for the case $k \neq l$. Therefore, the probabilities sum up to one.

The case for $k = l$ is similar, and we skip this case. □

To turn the PLPP into an LPP, one just needs to apply the construction in Lemma 15 of [30]. This completes the construction of an LPP that computes $\frac{\pi}{4}$.

4.3 Conclusion and Discussion

We relaxed the finitary restriction in Bournez et al. [3, 30] on fixed points in the ODE systems and showed that under the new definition, transcendental numbers like $\frac{\pi}{4}$ are LPP-computable. Although the LPP we constructed is specifically for computing $\frac{\pi}{4}$, the procedures are general enough for other applications. For example, by following similar procedures one can easily compute e^{-1}. Most steps of the procedure can be done by algorithms automatically once we have a PIVP, except for Step 3, in which we need to plan carefully what new species to introduce while we rewrite the PIVP into a degree-two polynomial. A way to automate Step 3 is much desired.

The ability to specify a set of marked variables makes something like addition very convenient. Does it add anything new in terms of computational power? Can we achieve the same thing without this ability?

Thus far, we have left out stability in our discussion. We now allow more than finitely many fixed points and they actually form a connected component of some algebraic variety. The exponential stability adopted in Bournez et al.'s is not suitable to our pursuit since the fixed points are no longer isolated.

The original definition of LPP-computable numbers proposed by Bournez et al. are closely related to the notion of Lyapunov CRN-computable numbers. Lyapunov-computable numbers are basically isolated fixed points. In a polynomial, there should be finitely many such points. One should be able to use a similar quantifier elimination method as in [3, 30] to show that Lyapunov CRN-computable numbers are just algebraic numbers. We will leave this topic to future research.

Another topic we left out was the rate of convergence. Bournez et al. gave a result on the rate of convergence in their work, but that depends heavily on the exponential stability of the fixed points, which is no longer the case here. Kurtz [31] also showed the limiting behavior of CRNs when the population becomes large. One might want to revisit his paper to find clues regarding the rate of convergence.

CHAPTER 5. ALGORITHMIC RANDOMNESS IN CONTINUOUS-TIME MARKOV CHAINS

5.1 Introduction

Stochastic chemical reaction networks are used in molecular programming, DNA nanotechnology, and synthetic biology to model and specify the behaviors of natural and engineered molecular systems. Stochastic chemical reaction networks are known to be Turing universal [54], hence capable of extremely complex dynamic behavior.

Briefly and roughly (deferring details until later in the chapter), a stochastic chemical reaction network N is a mathematical model of a chemical process in a volume V of solution. A state of N consists of the nonnegative integer populations of each of its finitely many species (types of molecules) at a given time. The state space is thus countable and discrete. The network stays in a state for a positive, real-valued sojourn time after which one of the finitely many reactions that N allows to occur among its species produces an instantaneous jump transition to a different state. Both the sojourn time and the choice of the reaction are probabilistic, with the network behaving as a certain kind of continuous-time Markov chain given by the parameters of N. Hence, given an initial state at time $t = 0$, there are typically many–perhaps uncountably many–trajectories (sequences of states and sojourn times) that N can traverse. Some of these trajectories are finite (because N reaches a state in which none of its reactions can occur), and some are infinite.

In this chapter we develop the elements of the theory of algorithmic randomness in continuous-time Markov chains (CTMCs). Specifically, our main contribution is a rigorous, useful notion of what it means for an *individual trajectory* (also called a single orbit) of a CTMC C to be *random* with respect to C and an initial state–or probability distribution of initial states–of C. This is a first step toward carrying out *Kolmogorov's program* of replacing probabilistic laws stating that *almost every* trajectory has a given property with randomness laws stating that *every random* trajectory has

the property. More generally, we are initiating an algorithmic "single orbit" approach (in the sense of Weiss [65]) to the dynamics of CTMCs. In a variety of contexts ranging from Bernoulli processes to ergodic theory, Brownian motion, and algorithmic learning, this algorithmic single-orbit approach has led to improved understanding of known results [35, 11, 44, 52, 62, 43, 15, 28, 2, 16, 60, 50, 17, 61]. In the context of fractal geometry, this approach has even led to recent solutions of classical open problems whose statements did not involve algorithms or single orbits [38, 39].

The fact that CTMCs have discrete state spaces and operate in continuous time, together with the fact that trajectories may or may not halt, presents challenges not encountered in more conventional developments of algorithmic randomness. Our formulation of randomness is nevertheless general. Because we are interested in the *computational* power of stochastic chemical reaction networks, we embrace situations in which the long-term behavior of a network depends essentially on its initial state. Our development thus does not make assumptions that are frequently used in Markov chain theory to avoid such dependencies.

Our approach is also general in another sense, one involving Kolmogorov's program, mentioned above. Once one has succeeded in replacing an "almost every" probabilistic law with an "every random" law, a natural next question is, "*How much* randomness is sufficient for the latter?" Saying that an individual object is random is saying that it "appears random" to a class of computations. Roughly speaking, an object is algorithmically random (or Martin-Löf random) if it appears random to all computably enumerable sets. But weaker notions of randomness such as computable randomness, polynomial-space randomness, polynomial-time randomness, and finite-state randomness, have also been extensively investigated. Three examples of answers to the "how much randomness suffices" question in the context of infinite binary sequences are that (i) every algorithmically random sequence satisfies Birkhoff's ergodic theorem [62]; (ii) every polynomial-time random sequence satisfies the Khinchin-Kolmogorov law of the iterated logarithm [63]; and (iii) every finite-state random sequence satisfies the strong law of large numbers [49].

Although we are primarily concerned with algorithmic randomness in the present chapter, we want our randomness notion to be general enough to extend easily to other computational''levels" of

randomness, so that "how much randomness" questions can be formulated and hopefully answered. For this reason, we define algorithmic randomness in CTMCs using the martingale (betting strategy) approach of Schnorr [47]. This approach extends to other levels of randomness in a straightforward manner, while our present state (i.e., lack) of knowledge in computational complexity theory does not allow us to extend other approaches (e.g., Martin-Löf tests or Kolmogorov complexity, which are known to be equivalent to the martingale approach at the algorithmic level [35, 11, 44, 52]) to time-bounded complexity classes.

We develop our algorithmic randomness theory in stages. In section 2 we develop the underlying qualitative structure of *Boolean transition systems*, defined so that (i) state transitions are nontrivial, i.e., not from a state to itself, and (ii) trajectories may or may not terminate. We then show how to use these transition systems to model rate-free chemical reaction networks.

In section 3 we add probabilities, thereby defining *probabilistic transition systems*. For each probabilistic transition system $\mathscr{Q}$ and each initialization σ of $\mathscr{Q}$ we then define $(\mathscr{Q}, \sigma)$-*martingales*, which are strategies for betting on the successive entries in a sequence of states of $(\mathscr{Q}, \sigma)$. Following the approach of Schnorr [47], we then define a maximal state sequence q of $(\mathscr{Q}, \sigma)$ to be *random* if there is no lower semicomputable $(\mathscr{Q}, \sigma)$-martingale that *succeeds* on q, i.e., makes unbounded money betting along q. This notion of randomness closely resembles the well-understood theory of random sequences over a finite alphabet [35, 11, 44, 52], except that here the state set may be countably infinite; transitions from a state to itself are forbidden; and a state sequence may terminate, in which case it is random.

Section 4 is where we confront the main challenge of algorithmic randomness in CTMCs, the fact that they operate in continuous, rather than discrete, time. There we develop the algorithmic randomness of sequences $\mathbf{t} = (t_0, t_1, ...)$ of sojourn times t_i relative to corresponding sequences $\boldsymbol{\lambda} = (\lambda_0, \lambda_1, ...)$ of nonnegative real-valued rates λ_i. Each λ_i in such a sequence is regarded as defining an exponential probability distribution function F_{λ_i}, and the sojourn times t_i are to be independently random relative to these. We use a careful binary encoding of sojourn times to define $\boldsymbol{\lambda}$-*martingales* that bet along sequences of sojourn times, and we again follow the Schnorr

approach, defining a sequence t of sojourn times to be $\boldsymbol{\lambda}$-*random* if there is no lower semicomputable $\boldsymbol{\lambda}$-martingale that succeeds in it.

In section 5 we put the developments of sections 3 and 4 together. A *trajectory* of a continuous-time Markov chain C is a sequence $\boldsymbol{\tau}$ of ordered pairs (q_n, t_n), where q_n is a state of C and t_n is the sojourn time that C spends in state q_n before jumping to state q_{n+1}. For each continuous-time Markov chain C, we define the notion of a *C-martingale*. Following Schnorr once again, we define a trajectory $\boldsymbol{\tau}$ of C to be *random* if no lower semicomputable martingale succeeds on it. As an example application we then prove that, in any stochastic chemical reaction network, *every* random trajectory $\boldsymbol{\tau}$ with bounded molecular counts has the *non-Zeno property* that infinitely many reactions do not occur in any finite interval of time. We also give a Kolmogorov complexity characterization of the randomness of trajectories of continuous-time Markov chains.

5.2 Boolean transition systems

Before developing algorithmic randomness for sequences of states with respect to computable, probabilistic transition systems, we develop the underlying qualitative (not probabilistic) structure by considering transition systems that are Boolean. Some care must be taken to accommodate the fact that, in cases of interest, a sequence of states may either be infinite or end in a terminal state.

Formally, we define a *Boolean transition system* to be an ordered pair $\mathscr{Q} = (Q, \delta)$ where Q is a nonempty, countable set of *states*, and $\delta : Q \times Q \to \{0, 1\}$ is a *Boolean state transition matrix* satisfying $\delta(q, q) = 0$ for all $q \in Q$.

Intuitively, a Boolean transition system $\mathscr{Q} = (Q, \delta)$ is a nondeterministic structure that may be initialized to any nonempty set of states in Q. For $q, r \in Q$, the entry $\delta(q, r)$ in the Boolean transition matrix δ is the Boolean value (0 = false; 1 = true) of the condition that r is reachable from q in one "step" of $\mathscr{Q}$. The irreflexivity requirement that every $\delta(q, q) = 0$ (i.e., that δ have a zero diagonal) reflects the fact that, in all cases of interest in this chapter, transitions are nontrivial changes of state. We formalize this intuition, because the formalism will be useful here.

We write $Q^{<\omega}$ for the set of all finite sequences of states in Q, Q^{ω} for the set of all infinite sequences of states in Q, and $Q^{\leq\omega} = Q^{<\omega} \cup Q^{\omega}$. The *length* of a sequence $\boldsymbol{q} \in Q^{\leq\omega}$ is

$$|\boldsymbol{q}| = \left\{ \begin{array}{ll} l & \textit{if } \boldsymbol{q} = (q_0, q_1, ..., q_{l-1}) \in Q^{<\omega} \\ \omega & \textit{if } \boldsymbol{q} \in Q^{\omega} \end{array} \right\}.$$

A sequence $\boldsymbol{q} \in Q^{\leq\omega}$ can thus be written as $\boldsymbol{q} = (q_i | i < |\boldsymbol{q}|)$ in any case. We write () for the *empty sequence* (sequence of length 0).

For $\boldsymbol{q}, \boldsymbol{r} = (r_i | i < |\boldsymbol{r}|) \in Q^{\leq\omega}$, we say that $\boldsymbol{q}$ is a *prefix* of $\boldsymbol{r}$, and we write $\boldsymbol{q} \sqsubseteq \boldsymbol{r}$, if $|\boldsymbol{q}| \leq |\boldsymbol{r}|$ and $\boldsymbol{q} = (r_i | i < |\boldsymbol{q}|)$. It is easy to see that $\sqsubseteq$ is a partial ordering of $Q^{\leq\omega}$.

An *initialization* of a Boolean transition system $\mathscr{Q} = (Q, \delta)$ is a Boolean-valued function $\sigma : Q \to \{0, 1\}$ whose *support* $supp(\sigma) = \{q \in Q | \sigma(q) \neq 0\}$ is nonempty.

A Boolean transition system $\mathscr{Q} = (Q, \delta)$ *admits* a sequence $\boldsymbol{q} = (q_i | i < |\boldsymbol{q}|) \in Q^{\leq\omega}$ with an initialization σ, and we say that $\boldsymbol{q}$ is *$\mathscr{Q}$-admissible* from σ, if the following conditions hold for all $0 \leq i < |\boldsymbol{q}|$.

(i) If $i = 0$, then $\sigma(q_i) = 1$.

(ii) If $i + 1 < |\boldsymbol{q}|$, then $\delta(q_i, q_{i+1}) = 1$.

A sequence $\boldsymbol{q} \in Q^{\leq\omega}$ that is $\mathscr{Q}$-admissible from σ is *maximal* if, for every sequence $\boldsymbol{r} \in Q^{\leq\omega}$ that is $\mathscr{Q}$-admissible from σ, $\boldsymbol{q} \sqsubseteq \boldsymbol{r} \implies \boldsymbol{q} = \boldsymbol{r}$.

We use the following notations.

$$Adm_{\mathscr{Q}}(\sigma) = \{\boldsymbol{x} \in Q^{<\omega} | \boldsymbol{x} \text{ is } \mathscr{Q}\text{-admissible from } \sigma\}$$

$$\mathbb{A}[\mathscr{Q}](\sigma) = \{\boldsymbol{q} \in Q^{\leq\omega} | \boldsymbol{q} \text{ is a maximal } \mathscr{Q}\text{-admissible sequence from } \sigma\}.$$

When $\mathscr{Q}$ is obvious from the context, we omit it from the notation and write these sets as $Adm(\sigma)$ and $\mathbb{A}(\sigma)$. Note that elements of $Adm_{\mathscr{Q}}(\sigma)$ are required to be *finite* sequences.

Intuitively, $\mathbb{A}[\mathscr{Q}](\sigma)$ is the set of all possible "behaviors" of the Boolean transition system $\mathscr{Q} = (Q, \delta)$ with the state initialization $\sigma : Q \to \{0, 1\}$. The fact that δ is irreflexive implies that $q_i \neq q_{i+1}$ holds for all $i \in \mathbb{N}$ such that $i + 1 < |\boldsymbol{q}|$ in every admissible sequence $\boldsymbol{q} = (q_i | i < |\boldsymbol{q}|) \in \mathbb{A}[\mathscr{Q}](\sigma)$. In

this chapter we do *not* regard the indices $i = 0, 1, ...$ in a state sequence $\boldsymbol{q} = (q_0, q_1, ...)$ as successive instants in discrete time. In our main applications, the amount of time spent in state q_i varies randomly and continuously, so it is more useful to think of the indices $i = 0, 1, ...$ as finite ordinal numbers, i.e., to think of q_i as merely the i^{th} state in the sequence $\boldsymbol{q}$.

Each $\boldsymbol{x} \in Adm_{\mathscr{Q}}(\sigma)$ is the *name* of the $\mathscr{Q}$*-cylinder*

$$\mathbb{A}_{\boldsymbol{x}}(\sigma) = \{\boldsymbol{q} \in \mathbb{A}[\mathscr{Q}](\sigma) | \boldsymbol{x} \sqsubseteq \boldsymbol{q}\}.$$

Each $\boldsymbol{x} \in Adm(\sigma)$ is a finite - and typically partial - specification of each sequence $\boldsymbol{q} \in \mathbb{A}_{\boldsymbol{x}}(\sigma)$. The collection

$$\mathscr{A}(\sigma) = \mathscr{A}[\mathscr{Q}](\sigma) = \{\mathbb{A}_{\boldsymbol{x}}(\sigma) | \boldsymbol{x} \in Adm_{\mathscr{Q}}(\sigma)\}$$

is a basis for a topology on $\mathbb{A}(\sigma)$. The open sets in this topology are simply the sets that are unions of (finitely or infinitely many) cylinders in $\mathscr{A}(\sigma)$. The metric (in fact, ultrametric) d on $Q^{\leq\omega}$ defined by

$$d(\boldsymbol{q}, \boldsymbol{r}) = 2^{-|\boldsymbol{p}|},$$

where $\boldsymbol{p}$ is the longest common prefix of $\boldsymbol{q}$ and $\boldsymbol{r}$ (and $2^{-\infty} = 0$), induces this same topology on $\mathbb{A}[\mathscr{Q}](\sigma)$ for each Boolean transition system $\mathscr{Q} = (Q, \delta)$ and each state initialization $\sigma : Q \to [0, 1]$. With this topology, $\mathbb{A}[\mathscr{Q}](\sigma)$ is a Polish space (a complete, separable metric space). The isolated points in $\mathbb{A}[\mathscr{Q}](\sigma)$ are (when they exist) the sequences in $\mathbb{A}[\mathscr{Q}](\sigma)$ that are finite, i.e., the sequences $x \in Q^{<\omega} \cap \mathbb{A}[\mathscr{Q}](\sigma)$. Such sequences $\boldsymbol{x}$ are said to *halt*, or *terminate*, in $\mathscr{Q}$ from σ.

A Boolean transition system $\mathscr{Q} = (Q, \delta)$ is *computable* if the elements of Q are naturally represented in such a way that (i) the Boolean-valued function δ is computable, and (ii) the set of *terminal states* (i.e., states $q \in Q$ such that $\delta(q, r) = 0$ for all $r \in Q$) is decidable. An initialization $\sigma : Q \to \{0, 1\}$ is *computable* if its support is decidable.

An important class of examples of Boolean transition systems consists of those that model rate-free chemical reaction networks. Formally, let $\boldsymbol{S} = \{X_0, X_1, X_2, ...\}$ be a countable set of distinct *species* X_n, each of which we regard as an abstract type of molecule. A *rate-free chemical reaction network* (or *rate-free CRN*) is an ordered pair $N = (S, R)$, where $S \subseteq \boldsymbol{S}$ is a finite set of

species, and R is a finite set of (*rate-free*) *reactions* on S, each of which is formally an ordered pair $\rho = (r, p)$ of distinct vectors $r, p \in \mathbb{N}^S$ (equivalently, functions $r, p : S \to \mathbb{N}$). Informally, we write species in notations convenient for specific problems ($X, Y, Z, \widehat{X}, \overline{Y}$, etc.) rather than as subscripted elements of $\boldsymbol{S}$, and we write reactions in a notation more suggestive of chemical reactions. For example,

$$X + Z \to 2Y + Z \tag{2.1}$$

is a rate-free reaction on the set $S = \{X, Y, Z\}$. If we consider the elements of S to be ordered as written, then the left-hand side of (2.1) is formally the *reactant vector* $r = (1, 0, 1)$, and the right-hand side of (2.1) is the *product vector* $p = (0, 2, 1)$. A species $Y \in S$ is called a *reactant* of a reaction $\rho = (r, p)$ if $r(Y) > 0$ and a *product* of ρ if $p(Y) > 0$.

Intuitively, the reaction ρ in (2.1) means that, if a molecule of species X encounters a molecule of species Z, then the reaction ρ may occur, in which case the reactants X and Z disappear and the products – two molecules of species Y and a molecule of species Z - appear in their place. Accordingly, the *net effect* of a reaction $\rho = (r, p)$ is the vector $\Delta\rho \in \mathbb{Z}^S$ defined by

$$\Delta\rho(Y) = p(Y) - r(Y) \tag{2.2}$$

for all $Y \in S$. Since we have required r and p to be distinct, $\Delta\rho$ is never the zero-vector in $\mathbb{Z}^S$.

In this chapter, a *state* of a chemical reaction network $N = (S, R)$ is a vector $q \in \mathbb{N}^S$. Intuitively, N is modeling chemical processes in a solution, and the state q denotes a situation in which, for each $Y \in S$, exactly $q(Y)$ molecules of species Y are present in the solution.

A reaction $\rho = (r, p) \in R$ of a chemical reaction network $N = (S, R)$ *can occur* in a state $q \in \mathbb{N}^S$ if

$$q(Y) \geq r(Y) \tag{2.3}$$

holds for every $Y \in S$, i.e. if the reactants of ρ are present in q. If this reaction ρ *does* occur in state q, then it transforms q to the new state $q + \Delta\rho$.

The behavior of a rate-free chemical reaction network $N = (S, R)$ clearly coincides with that of the Boolean transition system $\mathscr{Q}_N = (\mathbb{N}^S, \delta)$, where $\delta : \mathbb{N}^S \times \mathbb{N}^S \to [0, 1]$ is defined by setting

each $\delta(q, q')$ to be the Boolean value of the condition that some reaction $\rho \in R$ transforms the state q to the state q'. Boolean transition systems of this form are clearly computable and have other special properties. As one example, for each $q \in \mathbb{N}^S$, there only exist finitely many $q' \in \mathbb{N}^S$ for which $\delta(q, q') = 1$.

Rate-free chemical reaction networks, and Boolean transition systems more generally, raise significant and deep problems in distributed computing [32, 10], but our focus here is on randomness, which we begin in the following section.

5.3 Random state sequences

This section develops the elements of algorithmic randomness for sequences of states with respect to computable, probabilistic transition rules.

Formally, we define a *probabilistic transition system* to be an ordered pair $\mathscr{Q} = (Q, \pi)$, where Q is a countable set of *states*, and $\pi : Q \times Q \to [0, 1]$ is a *probabilistic transition matrix*, by which we mean that π satisfies the following two conditions for each state $q \in Q$.

(1) $\pi(q, q) = 0$.

(2) The sum $\pi(q) = \sum_{r \in Q} \pi(q, r)$ is either 0 or 1.

If the sum $\pi(q)$ in condition 2 is 0, then q is a *terminal state*. If $\pi(q)$ is 1, then q is a *nonterminal state*.

If $\mathscr{Q} = (Q, \pi)$ is a probabilistic transition system, and we define $\delta : Q \times Q \to \{0, 1\}$ by

$$\delta(q, r) = sgn(\pi(q, r))$$

for all $q, r \in Q$, where $sgn : [0, \infty) \to \{0, 1\}$ is the signum function

$$sgn(x) = \left\{ \begin{array}{ll} 0 & \textit{if } x = 0 \\ 1 & \textit{if } x > 0 \end{array} \right\},$$

then $\mathscr{Q}_B = (Q, \delta)$ is the Boolean transition system *corresponding* to Q. The essential difference between $\mathscr{Q}_B$ and $\mathscr{Q}$ is that, while $\delta(q, r)$ merely says whether it is *possible* for $\mathscr{Q}_B$ (or $\mathscr{Q}$) to transition from q to r in one step, $\pi(q, r)$ is the *quantitative probability* of doing so.

An *initialization* of a probabilistic transition system $\mathscr{Q} = (Q, \pi)$ is a discrete probability measure σ on Q, i.e., a function $\sigma : Q \to [0, 1]$ satisfying $\sum_{q \in Q} \sigma(q) = 1$. The *Boolean version* of such an initialization σ is the function $\sigma_B : Q \to \{0, 1\}$ defined by

$$\sigma_B(q) = sgn(\sigma(q))$$

for each $q \in Q$. It is clear that σ_B is an initialization of $\mathscr{Q}_B$.

Given a probabilistic transition system $\mathscr{Q} = (Q, \pi)$ and an initialization σ of $\mathscr{Q}$, we define the sets

$$Adm(\sigma) = Adm_{\mathscr{Q}}(\sigma) = Adm_{\mathscr{Q}_B}(\sigma_B),$$
$$\mathbb{A}(\sigma) = \mathbb{A}[\mathscr{Q}](\sigma) = \mathbb{A}_{\mathscr{Q}_B}(\sigma_B),$$

relying on the fact that the right-hand sets were defined in section 2. The notations and terminology in section 2 leading up to these definitions are similarly extended to probabilistic transition systems, as are the definitions of the $\mathscr{Q}$-cylinders $\mathbb{A}_{\boldsymbol{x}}(\sigma)$ and the basis $\mathscr{A}(\sigma)$ for the topology $\mathbb{A}(\sigma)$.

What we can do here that we could not do for Boolean transition systems is define a Borel probability measure on each set $\mathbb{A}[\mathscr{Q}](\sigma)$. Specifically, for each probabilistic transition system $\mathscr{Q} = (Q, \pi)$ and each initialization σ of $\mathscr{Q}$, define the function

$$\mu_{\mathscr{Q},\sigma} : Adm_{\mathscr{Q}}(\sigma) \to [0, 1]$$

as follows. Let $\boldsymbol{x} = (x_i | i < |\boldsymbol{x}|) \in Adm_{\mathscr{Q}}(\sigma)$. If $|\boldsymbol{x}| = 0$, then $\mu_{\mathscr{Q},\sigma}(x) = 1$. If $|\boldsymbol{x}| > 0$, then

$$\mu_{\mathscr{Q},\sigma}(\boldsymbol{x}) = \sigma(x_0) \prod_{i=0}^{|\boldsymbol{x}|-2} \pi(x_i, x_{i+1}). \tag{3.1}$$

Since $\boldsymbol{x}$ is a name of the cylinder $\mathbb{A}_{\boldsymbol{x}}[\mathscr{Q}](\sigma)$, each $\mu_{\mathscr{Q},\sigma}(\boldsymbol{x})$ here should be understood as an abbreviation of $\mu_{\mathscr{Q},\sigma}(\mathbb{A}_{\boldsymbol{x}}(\sigma))$, which is intuitively the probability that an element of $\mathbb{A}_{\boldsymbol{x}}[\mathscr{Q}](\sigma)$ begins with the finite sequence $\boldsymbol{x}$.

Observation 5.3.1. *If a sequence $\boldsymbol{x} \in Adm_{\mathscr{Q}}(\sigma)$ does not terminate, then*

$$\mu_{\mathscr{Q},\sigma}(\boldsymbol{x}) = \sum_{\boldsymbol{x} \sqsubseteq \boldsymbol{y} \in Adm_{\mathscr{Q}}(\sigma), |\boldsymbol{y}| = |\boldsymbol{x}|+1} \mu_{\mathscr{Q},\sigma}(\boldsymbol{y}) \tag{3.2}$$

The above observation implies that $\mu_{\mathscr{Q},\sigma}$ can, by standard techniques, be extended to a Borel probability measure on $\mathbb{A}[\mathscr{Q}](\sigma)$, i.e., to a function $\mu_{\mathscr{Q},\sigma}$ that assigns probability $\mu_{\mathscr{Q},\sigma}(E)$ to every Borel set $E \subseteq \mathbb{A}[\mathscr{Q}](\sigma)$.

Definition 3. *If $\mathscr{Q}$ is a probabilistic transition system and σ is an initialization of $\mathscr{Q}$, then a $(\mathscr{Q},\sigma)$-martingale is a function*

$$d : Adm_{\mathscr{Q}}(\sigma) \to [0,\infty)$$

such that, for every non-terminating sequence $\boldsymbol{x} \in Adm_{\mathscr{Q}}(\sigma)$,

$$d(\boldsymbol{x})\mu(\boldsymbol{x}) = \sum_{\boldsymbol{x} \sqsubseteq \boldsymbol{y} \in Adm_{\mathscr{Q}}(\sigma), |\boldsymbol{y}| = |\boldsymbol{x}|+1} d(\boldsymbol{y})\mu_{\mathscr{Q},\sigma}(\boldsymbol{y}) \tag{3.3}$$

where $\mu = \mu_{\mathscr{Q},\sigma}$.

Intuitively, a $(\mathscr{Q},\sigma)$-martingale d is a gambler that bets on the successive states in a sequence $\boldsymbol{q} = (q_i | i < |\boldsymbol{q}|) \in \mathbb{A}[\mathscr{Q}](\sigma)$. The gambler's initial capital is $d(())$, and its capital after betting on a prefix $x \in Adm_{\mathscr{Q}}(\sigma)$ of $\boldsymbol{q}$ is $d(\boldsymbol{x})$. The condition (3.3) says that the payoffs are fair with respect to the probability measure $\mu = \mu_{\mathscr{Q},\sigma}$ in the sense that the conditional expectation of the gambler's capital after betting on the state following $\boldsymbol{x}$ in $\boldsymbol{q}$ given that $\boldsymbol{x} \sqsubseteq \boldsymbol{q}$, is exactly the gambler's capital before placing this bet.

Definition 4. *A $(\mathscr{Q},\sigma)$-martingale d succeeds on a sequence $\boldsymbol{q} \in \mathbb{A}[\mathscr{Q}](\sigma)$ if the set*

$$\{d(\boldsymbol{x}) | \boldsymbol{x} \in Adm_{\mathscr{Q}}(\sigma) \text{ and } \boldsymbol{x} \sqsubseteq \boldsymbol{q}\}$$

is unbounded.

The success set of a $(\mathscr{Q},\sigma)$-martingale d is $S^{\infty}[d] = \{\boldsymbol{q} \in \mathbb{A}[\mathscr{Q}](\sigma) | d \text{ succeeds on } \boldsymbol{q}\}$.

Following standard practice, we develop randomness by imposing computability conditions on martingales. Recall that, if D is a discrete domain, then a function $f : D \to \mathbb{R}$ is *computable* if there is a computable function $\widehat{f} : D \times \mathbb{N} \to \mathbb{Q}$ such that, for all $x \in D$ and $r \in \mathbb{N}$,

$$|\widehat{f}(x,r) - f(x)| \leq 2^{-r}.$$

The parameter r here is called a *precision parameter*.

A function $f : D \to \mathbb{R}$ is *lower semi-computable* if there is a computable function $\widehat{f} : D \times \mathbb{N} \to \mathbb{Q}$ such that the following two conditions hold for all $x \in D$.

(i) For all $s \in \mathbb{N}$, $\widehat{f}(x, s) \leq \widehat{f}(x, s+1) < f(x)$.

(ii) $\lim_{s\to\infty} \widehat{f}(x, s) = f(x)$.

The parameter s is sometimes called a *patience parameter*, because the convergence in (ii) can be very slow. A probabilistic transition system $\mathscr{Q} = (Q, \pi)$ is *computable* if the elements of Q are naturally represented in such a way that (i) the probability transition matrix $\pi : Q \times Q \to [0, 1]$ is computable in the above sense, and (ii) the support of π and the set of terminal states are decidable. (It is well known ([29], [64]) that (ii) does not follow from (i). Fortunately, (ii) does hold in many cases of interest, including chemical reaction networks).

Similarly, an initialization σ of a probabilistic transition system $\mathscr{Q} = (Q, \pi)$ is *computable* if (i) the function $\sigma : Q \to [0, 1]$ is computable, and (ii) the *support* of σ is decidable.

Let $\mathscr{Q}$ be a probabilistic transition system that is computable, and let σ be an initialization of $\mathscr{Q}$ that is also computable. A state sequence $\boldsymbol{q} \in \mathbb{A}[\mathscr{Q}](\sigma)$ is (*algorithmically*) *random* if there is no lower semi-computable $(\mathscr{Q}, \sigma)$-martingale that succeeds on $\boldsymbol{q}$.

This notion of random sequences in $\mathbb{A}[\mathscr{Q}](\sigma)$ closely resembles the well-understood theory of random sequences on a finite alphabet [68, 48]. The main differences are that here the state set may be countably infinite; transitions from a state to itself are forbidden; and a state sequence may terminate, in which case it is clearly random. The following analogue of Ville's theorem holds for probabilistic transition sequences. It is proven in Section 5.

Theorem 1 (Ville [59])**.** *Let $\mathscr{Q}$ be a probabilistic transition system, let σ be an initialization of $\mathscr{Q}$, and let $\mu = \mu_{\mathscr{Q},\sigma}$. For every set $E \subseteq \mathbb{A}[\mathscr{Q}](\sigma)$, the following two conditions are equivalent.*

(1) $\mu(E) = 0$.

(2) There is a $(\mathscr{Q}, \sigma)$-martingale d such that $E \subseteq S^{\infty}[d]$.

5.4 Random sequences of sojourn times

The "sojourn time" that a continuous-time Markov chain spends in a state before jumping to a new state may be any element of $(0, \infty]$, i.e., any duration t that is either a (strictly) positive real number or ∞. This section thus develops the elements of algorithmic randomness for sequences of durations $t \in (0, \infty]$ with respect to sequences of probability measures that occur in continuous-time Markov chains.

A *rate* in this chapter is a non-negative real number $\lambda \in [0, \infty)$. We rely on context to distinguish this standard use of λ from the equally standard use of λ to denote the empty string.

We interpret each rate $\lambda > 0$ as a name of the exponential probability measure with rate λ, i.e., the probability measure on $(0, \infty]$ whose cumulative distribution function $F_\lambda : (0, \infty] \to [0, 1]$ is given by

$$F_\lambda(t) = 1 - e^{-\lambda t}$$

for all $t \in (0, \infty]$, where $e^{-\infty} = 0$. We interpret the rate $\lambda = 0$ as a name of the point-mass probability on $(0, \infty]$ that concentrates all the probability at ∞. This has the cumulative distribution function $F_0 : (0, \infty] \to [0, 1]$ given by

$$F_0(t) = \left\{ \begin{array}{ll} 0 & \text{if } t \in (0, \infty) \\ 1 & \text{if } t = \infty \end{array} \right\}$$

We associate each string $w \in \{0, 1\}^*$ with the interval $I_w \subseteq [0, 1]$ defined as follows. Let w be the lexicographically i^{th} $(0 \leq i < 2^{|w|})$ element of $\{0, 1\}^{|w|}$ where $0^{|w|}$ is the 0^{th} element and $1^{|w|}$ is the $(2^{|w|} - 1)^{\text{st}}$ element. Then

$$I_w = (2^{-|w|} i, 2^{-|w|}(i + 1)].$$

Note that, for each $w \in \{0, 1\}^*$ and $l \in \mathbb{N}$, the intervals I_{wu}, for $u \in \{0, 1\}^l$, form a *left-to-right partition* of I_w, i.e., a partition of I_w in which I_{wu} lies to the left of I_{wv} if and only if u lexicographically precedes v.

For each rate $\lambda \in [0, \infty)$ and each string $w \in \{0, 1\}^*$, define the interval

$$D_\lambda(w) = F_\lambda^{-1}(I_w) \subseteq (0, \infty].$$

Example 5.4.1. *If $\lambda > 0$, then*

$$D_\lambda(00) = (0, a_1], \quad D_\lambda(01) = (a_1, a_2],$$
$$D_\lambda(10) = (a_2, a_3], \quad D_\lambda(11) = (a_3, \infty],$$

where $a_1 = \frac{2ln2-ln3}{\lambda}, a_2 = \frac{ln2}{\lambda}$, and $a_3 = \frac{2ln2}{\lambda}$. On the other hand, $D_0(00) = (0, \infty)$, $D_0(01) = D_0(10) = \emptyset$, and $D_0(11) = \{\infty\}$.

Observation 5.4.2. *If $\lambda > 0$, then, for each $l \in \mathbb{N}$, the intervals $D_\lambda(w)$, for $w \in \{0,1\}^l$, form a left-to-right partition of $(0, \infty]$ into intervals that are equiprobable with respect to F_λ.*

Example 5.4.1 shows that the assumption $\lambda > 0$ is essential here.

For each rate $\lambda \in [0, \infty)$, each duration $t \in (0, \infty]$, and each $w \in \{0,1\}^*$, we call w a λ-*approximation* (or a *partial λ-specification*) of t, and we write $w \sqsubseteq_\lambda t$, if $t \in D_\lambda(w)$.

A *rate sequence* is a nonempty sequence $\boldsymbol{\lambda} = (\lambda_i \mid 0 \leq i < |\boldsymbol{\lambda}|) \in [0, \infty)^{\leq \omega}$ with the property that, for each $0 \leq i < |\boldsymbol{\lambda}|$,

$$i + 1 < |\boldsymbol{\lambda}| \iff \lambda_i > 0.$$

(That is, either $\boldsymbol{\lambda}$ is finite with a single 0 entry, occurring at the end, or $\boldsymbol{\lambda}$ is infinite with no 0 entries.)

If $\boldsymbol{\lambda} = (\lambda_i \mid 0 \leq i < |\boldsymbol{\lambda}|)$ is a rate sequence, then a $\boldsymbol{\lambda}$-*duration sequence* is a sequence

$$\boldsymbol{t} = (t_i | i < |\boldsymbol{\lambda}|) \in (0, \infty]^{\leq \omega}$$

such that, for each $0 \leq i < |\boldsymbol{\lambda}|$,

$$t_i < \infty \iff \lambda_i > 0.$$

We write $\boldsymbol{D_\lambda}$ for the set of all $\boldsymbol{\lambda}$-duration sequences. Note that

$$\boldsymbol{D_\lambda} = \left\{ \begin{array}{ll} (0, \infty)^{|\boldsymbol{\lambda}|-1} \times \{\infty\} & \text{if } |\boldsymbol{\lambda}| < \omega \\ (0, \infty)^\omega & \text{if } |\boldsymbol{\lambda}| = \omega \end{array} \right\}$$

depends only on the length of $\boldsymbol{\lambda}$, not on the components of $\boldsymbol{\lambda}$.

If $\boldsymbol{\lambda} = (\lambda_i \mid 0 \leq i < |\boldsymbol{\lambda}|)$ is a rate sequence, $\boldsymbol{t} = (t_i | i < |\boldsymbol{\lambda}|) \in \boldsymbol{D}_{\boldsymbol{\lambda}}$ is a $\boldsymbol{\lambda}$-duration sequence, and $\boldsymbol{w} = (w_i | i < |\boldsymbol{w}|) \in (\{0,1\}^*)^{<\omega}$ is a finite sequence of binary strings with $|\boldsymbol{w}| \leq |\boldsymbol{\lambda}|$, then we call $\boldsymbol{w}$ a $\boldsymbol{\lambda}$-*approximation* (or a *partial* $\boldsymbol{\lambda}$-*specification*) of $\boldsymbol{t}$, and we write $\boldsymbol{w} \sqsubseteq_{\boldsymbol{\lambda}} \boldsymbol{t}$, if $w_i \sqsubseteq_{\lambda_i} t_i$ holds for all $0 \leq i < |\boldsymbol{w}|$.

If $\boldsymbol{\lambda}$ is a rate sequence and $\boldsymbol{w} \in (\{0,1\}^*)^{<\omega}$ is a finite sequence of binary strings with $|\boldsymbol{w}| \leq |\boldsymbol{\lambda}|$, then the $\boldsymbol{\lambda}$-*cylinder generated by* $\boldsymbol{w}$ is the set

$$\boldsymbol{D}_{\boldsymbol{\lambda}}(\boldsymbol{w}) = \{\boldsymbol{t} \in \boldsymbol{D}_{\boldsymbol{\lambda}} \mid \boldsymbol{w} \sqsubseteq_{\boldsymbol{\lambda}} \boldsymbol{t}\}$$

of $\boldsymbol{\lambda}$-duration sequences.

It is routine to verify that, for each rate sequence $\boldsymbol{\lambda}$, the collection

$$\mathscr{D}_{\boldsymbol{\lambda}} = \{\boldsymbol{D}_{\boldsymbol{\lambda}}(\boldsymbol{w}) \mid \boldsymbol{w} \in (\{0,1\}^*)^{<\omega} \text{ and } |\boldsymbol{w}| \leq |\boldsymbol{\lambda}|\}$$

is a semi-algebra of subsets of $\boldsymbol{D}_{\boldsymbol{\lambda}}$ that generates the σ-algebra $\mathscr{B}_{\boldsymbol{\lambda}}$ of all Borel subsets of $\mathscr{D}_{\boldsymbol{\lambda}}$. If we define

$$\mu_{\boldsymbol{\lambda}} : \mathscr{D}_{\boldsymbol{\lambda}} \to [0,1]$$

by

$$\mu_{\boldsymbol{\lambda}}(\mathscr{D}_{\boldsymbol{\lambda}}(\boldsymbol{w})) = 2^{-\sum_{i=0}^{|\boldsymbol{w}|-1} |w_i|}$$

for all $\boldsymbol{w} = (w_i \mid i < |\boldsymbol{w}|) \in (\{0,1\}^*)^{<\omega}$ with $|\boldsymbol{w}| \leq |\boldsymbol{\lambda}|$, then it follows by standard techniques that $\mu_{\boldsymbol{\lambda}}$ extends uniquely to a probability measure

$$\mu_{\boldsymbol{\lambda}} : \mathscr{B}_{\boldsymbol{\lambda}} \to [0,1].$$

Note that $\mathscr{B}_{\boldsymbol{\lambda}}$ only depends on the length of $\boldsymbol{\lambda}$, but $\mu_{\boldsymbol{\lambda}}$ also depends on the components of $\boldsymbol{\lambda}$. When convenient, we use the abbreviation

$$\mu_{\boldsymbol{\lambda}}(\boldsymbol{w}) = \mu_{\boldsymbol{\lambda}}(\boldsymbol{D}_{\boldsymbol{\lambda}}(\boldsymbol{w})).$$

If $\boldsymbol{\lambda} = (\lambda_i \mid 0 \leq i < |\boldsymbol{\lambda}|)$ is a rate sequence, then a $\boldsymbol{\lambda}$-*martingale* is a function

$$d : (\{0,1\}^*)^{<|\boldsymbol{\lambda}|} \to [0,\infty)$$

that satisfies the following two conditions for all $\boldsymbol{w} = (w_0, ..., w_{n-1}) \in (\{0,1\}^*)^{<|\boldsymbol{\lambda}|}$.

1. $d(\boldsymbol{w}) = \frac{d(w_0,...,w_{n-1}0)+d(w_0,...,w_{n-1}1)}{2}$.

2. If $n+1 < |\boldsymbol{\lambda}|$, then

$$d(w_0, ..., w_{n-1}, \lambda) = d(w_0, ..., w_{n-1}).$$

(Note that the λ entry on the left-hand side is the empty string.)

Intuitively, a $\boldsymbol{\lambda}$-martingale d is a strategy that a gambler may use for betting on approximations w_i of the durations t_i in a $\boldsymbol{\lambda}$-duration sequence $\boldsymbol{t} = (t_i \mid i < |\boldsymbol{t}|)$. The gambler's initial amount of money is the value $d(())$ of d at the empty sequence () of binary strings. If $\boldsymbol{w} = (w_0, ..., w_{n-1}) \sqsubseteq_{\boldsymbol{\lambda}} \boldsymbol{t}$, then $d(\boldsymbol{w})$ is the amount of money that the gambler has after betting on $\boldsymbol{w}$. This condition $\boldsymbol{w} \sqsubseteq_{\boldsymbol{\lambda}} \boldsymbol{t}$ means that each t_i is in the interval $\boldsymbol{D}_{\boldsymbol{\lambda}}(w_i) \subseteq (0, \infty]$. If the gambler then chooses to bet on which of the subintervals $D_{\lambda_{n-1}}(w_{n-1}0)$ and $D_{\lambda_{n-1}}(w_{n-1}1)$ of $D_{\lambda_{n-1}}(w_{n-1})$ t_{n-1} lies in, condition 1 above says that the payoffs of these bets are fair with respect to the exponential probability measure with rate λ_{n-1}. (Note that $D_{\lambda_{n-1}}(w_{n-1}0)$ and $D_{\lambda_{n-1}}(w_{n-1}1)$ partition $D_{\lambda_{n-1}}(w_{n-1})$ into equiprobable subintervals, but these subintervals may have very different lengths.) Condition 2 above says that the extension from $(w_0, ..., w_{n-1})$ to $(w_0, ..., w_{n-1}, \lambda)$, does not involve a bet. The martingale has values $d(\boldsymbol{w})$ for all $\boldsymbol{w} \in (\{0,1\}^*)^{<|\boldsymbol{\lambda}|}$, but our intuitive gambler may place bets in many different orders. For example, the gambler may place a finite number of bets on approximations of t_1, then a finite number of bets on approximations of t_2, etc., but this ordering of bets is an intuitive fancy, not part of the definition of the $\boldsymbol{\lambda}$-martingale d.

A $\boldsymbol{\lambda}$-martingale d *succeeds* on a $\boldsymbol{\lambda}$-duration sequence $\boldsymbol{t}$ if the set

$$\{d(w) \mid \boldsymbol{w} \sqsubseteq_{\boldsymbol{\lambda}} \boldsymbol{t}\}$$

is unbounded. The *success set* of a $\boldsymbol{\lambda}$-martingale d is

$$S^{\infty}[d] = \{\boldsymbol{t} \in \boldsymbol{D}_{\boldsymbol{\lambda}} \mid d \text{ succeeds on } \boldsymbol{t}\}.$$

An analogue of Ville's theorem holds also for $\boldsymbol{\lambda}-$martingales. It is proven in Section 5.

Theorem 2. *If $\boldsymbol{\lambda}$ is a rate sequence, then, for each set $E \subseteq \boldsymbol{D}_{\boldsymbol{\lambda}}$, the following two conditions are equivalent.*

1. $\mu_{\boldsymbol{\lambda}}(E) = 0$
2. *There is a $\boldsymbol{\lambda}$-martingale d such that $E \subseteq S^{\infty}[d]$.*

5.5 Random CTMC trajectories

We now develop the theory of randomness for sequences of state-time pairs, representing trajectories of continuous-time Markov chains.

5.5.1 Continuous-time Markov chains

A CTMC is an ordered triple,

$$C = (Q, \boldsymbol{\lambda}, \sigma)$$

where Q is a countable set of states, $\boldsymbol{\lambda} : Q \times Q \to [0, \infty)$ is the *rate matrix* satisfying $\lambda(q, q) = 0$ for every $q \in Q$, and σ is the *state initialization* as described in section 3. Let $C = (Q, \boldsymbol{\lambda}, \sigma)$ be a CTMC. At each time $t \in [0, \infty)$ C is probabilistically in some state. At time $t = 0$, this state is chosen according to σ. For each state $q \in Q$, the real number

$$\lambda_q = \sum_{r \in Q} \lambda(q, r)$$

is the *rate out of* state q. If $\lambda_q = 0$, then q is a *terminal* state, meaning that, if C ever enters state q, then C remains in state q forever. If a state q is *nonterminal*, i.e., $\lambda_q > 0$ and C enters q at some time t, then the *sojourn time* for which C remains in state q before moving to a new state is a random variable that has the exponential distribution with rate λ_q. Hence the expected sojourn time of C in state q is $\frac{1}{\lambda_q}$. When C does move to a new state, it moves to state $r \in Q$ with probability

$$p(q, r) = \frac{\lambda(q, r)}{\lambda_q}.$$

Note that the CTMC model uses "continuous time" (times ranging over $(0, \infty]$) but "discrete state space". Accordingly, its state transitions, called *jump transitions*, are instantaneous. Mathematically,

if C jumps from state q to state r at time t, we say that q is in the "new" state r at time t, having been in the "old" state q throughout some time interval $[s, t)$ where $s < t$.

A *trajectory* of a CTMC $C = (Q, \boldsymbol{\lambda}, \sigma)$ is a sequence $\boldsymbol{\tau}$ of the form

$$\boldsymbol{\tau} = ((q_n, t_n) \mid n \in \mathbb{N}) \in (Q \times (0, \infty))^{\infty}.$$

Intuitively, such a trajectory $\boldsymbol{\tau}$ denotes the turn of events in which $q_0, q_1, ...$ are the successive states of C and $t_0, t_1, ...$ are the successive sojourn times of C in these states. Accordingly, we write

$$state_{\boldsymbol{\tau}}(n) = q_n, \ soj_{\boldsymbol{\tau}}(n) = t_n$$

for each $n \in \mathbb{N}$. When convenient we write $\boldsymbol{\tau}$ as an ordered pair

$$\boldsymbol{\tau} = (\boldsymbol{q}, \boldsymbol{t}),$$

where

$$\boldsymbol{q} = (q_n \mid n \in \mathbb{N}), \boldsymbol{t} = (t_n \mid n \in \mathbb{N}).$$

There are two ways in which a trajectory $(\boldsymbol{q}, \boldsymbol{t})$ may fail to represent a "true trajectory" of the CTMC C in the above intuitive sense. First, it may be the case that $p(q_n, q_{n+1}) = 0$ (i.e. $\lambda(q_n, q_{n+1}) = 0$) for some $n \in \mathbb{N}$. This presents no real difficulty, since it merely says that the event "$state_{\boldsymbol{\tau}}(n) = q_n$ and $state_{\boldsymbol{\tau}}(n+1) = q_{n+1}$" has probability 0. The second way in which $(\boldsymbol{q}, \boldsymbol{t})$ may fail to represent a "true trajectory" is for some q_n to be a terminal state of C. We deal with this by defining the *length* of a trajectory $\boldsymbol{\tau} = (\boldsymbol{q}, \boldsymbol{t})$ to be

$$||\boldsymbol{\tau}|| = \min\{n \in \mathbb{N} \mid q_n \text{ is terminal } \},$$

where $min\ \emptyset = \infty$. We then intuitively interpret a trajectory $\boldsymbol{\tau} = (\boldsymbol{q}, \boldsymbol{t})$ with $||\boldsymbol{\tau}|| < \infty$ as the finite sequence

$$\boldsymbol{\tau}' = ((q_n, t'_n) \mid n \leq ||\boldsymbol{\tau}||),$$

where each

$$t'_n = \begin{cases} t_n & \text{if } n < ||\boldsymbol{\tau}|| \\ \infty & \text{if } n = ||\boldsymbol{\tau}|| \end{cases} \tag{5.5.1}$$

We write

$$\Omega = \Omega[C] = (Q \times (0, \infty))^\infty$$

for the set of all trajectories of a CTMC, C.

Elements of $(Q \times \{0,1\}^*)^*$ are called *approximations* or *partial specifications* of trajectories. The *cylinder generated by* $w = (q_0, u_0), (q_1, u_1), ..., (q_{n-1}, u_{n-1}) \in (Q \times \{0,1\}^*)^*$ is the set Ω_w of trajectories defined as follows: If q_i is terminal for some $0 \leq i < n-1$ then $\Omega_w = \emptyset$. If q_i is nonterminal for all $0 \leq i < n-1$ and q_{n-1} is terminal, then

$$\Omega_w = \{\boldsymbol{\tau} \in \Omega \mid (\forall 0 \leq i < n) state_{\boldsymbol{\tau}}(i) = q_i$$

$$\text{and } (\forall 0 \leq i < n-1) soj_{\boldsymbol{\tau}}(i) \in D_{\lambda_i}(u_i)\}.$$

If q_i is nonterminal for all $0 \leq i < n$ then

$$\Omega_w = \{\boldsymbol{\tau} \in \Omega \mid (\forall 0 \leq i < n)[state_{\boldsymbol{\tau}}(i) = q_i$$

$$\text{and } soj_{\boldsymbol{\tau}}(i) \in D_{\lambda_i}(u_i)]\}.$$

The *probability* $\mu_C(\Omega_w)$, usually written $\mu_C(w)$, of a cylinder Ω_w, is defined as follows: If $n = 0$ (i.e. $w = \lambda$), then $\mu_C(w) = 1$. If q_i is terminal for some $0 \leq i < n-1$, then $\mu_C(\Omega_w) = 0$. If q_i is nonterminal for all $0 \leq i < n-1$ and q_{n-1} is terminal, then

$$\mu_C(\Omega_w) = \sigma(q_0) \prod_{i=0}^{n-2} [p(q_i, q_{i+1}) 2^{-|u_i|}].$$

If $n > 0$ and q_i is nonterminal for all $0 \leq i < n$, then

$$\mu_C(\Omega_w) = \sigma(q_0) \prod_{i=0}^{n-2} p(q_i, q_{i+1}) \prod_{i=0}^{n-1} 2^{-|u_i|}.$$

A set $X \subseteq \Omega$ has *probability 0*, and we write $\mu_C(X) = 0$, if, for $\epsilon > 0$, there is a set $A \subseteq (Q \times \{0,1\}^*)^*$ such that

$$X \subseteq \bigcup_{w \in A} \Omega_w$$

and

$$\sum_{w \in A} \mu_C(\Omega_w) \leq \epsilon$$

From now on we assume that the states $q \in Q$ have canonical representations, so that it is clear what it means for function $f : Q \to Q$, etc., to be computable.

A set $X \in \Omega$ has *constructive probability 0* (or is a *constructive null set*), and we write $\mu_{C,\text{constr}}(X) = 0$, if there is a computable function

$$g : \mathbb{N} \times \mathbb{N} \to (Q \times \{0,1\}^*)^*$$

such that, for every $k \in \mathbb{N}$,

$$X \subseteq \bigcup_{l=0}^{\infty} \Omega_{g(k,l)}$$

and

$$\sum_{l=0}^{\infty} \mu_C(\Omega_{g(k,l)}) \leq 2^{-k}.$$

A set $X \subseteq \Omega$ has *constructive probability 1*, and we write

$$\mu_{C,\text{constr}}(X) = 1,$$

if $\mu_{C,\text{constr}}(\Omega \setminus X) = 0$.

Before we discuss C-martingales and their relation to the above probability space, let us overload the 'prefix' relation $\sqsubseteq$ to compare partial specifications to partial specifications and to trajectories. If $w \in (Q \times \{0,1\}^{<\omega})^{<\omega}$ and $S \in (Q \times \{0,1\}^{\leq\omega})^{\leq\omega}$, we say $w \sqsubseteq S$ if:

1. $|v| \leq |w|$
2. For all i, $0 \leq i \leq |v| - 1, w[i] \sqsubseteq v[i]$ or $v[i] \sqsubseteq w[i]$
3. For all i, $0 \leq i \leq |v| - 1, \mathit{state}(w[i]) = \mathit{state}(v[i])$

Note that a trajectory can have two prefixes which are not prefixes of one another. Note also that two partial specifications (whether of real sequences or of trajectories) may prefix one another without being identical. In some ways, is more appropriate to call $\sqsubseteq$ in the context of sequences of times and in the context of trajectories a *disjointness relation* in the sense that

$$v \not\sqsubseteq w \text{ and } w \not\sqsubseteq v \implies \Omega_w \cap \Omega_v = \emptyset.$$

We now introduce the notion of a C-martingale.

5.5.2 CTMC martingales

In place of $\mu_{\boldsymbol{\lambda}}, \mu_C$, and $\mu_{\mathscr{Q},\sigma}$ we will simply write μ, μ_{constr}. It should be clear from context which measure is being used.

If $C = (Q, \boldsymbol{\lambda}, \pi)$ is a CTMC, then a *C-martingale* is a function

$$d : (Q \times \{0,1\}^*)^* \to [0, \infty)$$

with the following two properties.

1. For all $w \in (Q \times \{0,1\}^*)^*$,

$$d(w)\mu(w) = \sum_{q \in Q} d(w(q,\lambda))\mu(w(q,\lambda)). \tag{5.5.2}$$

2. For all $w \in (Q \times \{0,1\}^*)^*$, $q \in Q$, and $u \in \{0,1\}^*$,

$$d(w(q,u))\mu(w(q,u)) = \sum_{b \in \{0,1\}} d(w(q,ub))\mu(w(q,ub)) \tag{5.5.3}$$

Intuitively, a C-martingale d is a strategy for betting on successive approximations w of a trajectory $\boldsymbol{\tau}$ of C. A gambler using d starts with initial capital $d(\lambda) \in [0, \infty)$. More generally, each value $d(w)$ is the amount of money that the gambler will have after betting on w. At this stage, the C-martingale d tells the gambler how it may proceed in either of the following two ways.

(i) The gambler may "move on" to bet on the value of $state_{\boldsymbol{\tau}}(|w|)$, which is the next state of $\boldsymbol{\tau}$. In this case condition (5.5.2) ensures that the payoffs for this bet are fair.

(ii) The gambler may "stay" with the current state, which is $state_{\boldsymbol{\tau}}(|w|-1)$, and bet further on the approximate value of $soj_{\boldsymbol{\tau}}(|w|-1)$. In this case condition (5.5.3) ensures that the payoffs for this bet are fair.

A C-martingale d *succeeds* on a trajectory $\boldsymbol{\tau}$ if, for every real number $\alpha > 0$, there exists $w \in (Q \times \{0,1\}^*)^*$ such that $w \sqsubseteq \boldsymbol{\tau}$ and $d(w) > \alpha$.

The *success set* of a C-martingale d is

$$S^{\infty}[d] = \{\boldsymbol{\tau} \in \Omega[C] \mid \text{d succeeds on } \boldsymbol{\tau}\}.$$

An analogue of Ville's theorem holds for $C-$martingales. In order to prove this, we first prove two useful lemmas.

Lemma 3 (Generalized Kraft Inequality)**.** *Let $C = (Q, \boldsymbol{\lambda}, \sigma)$ be a CRN, d a C-martingale (resp. $\boldsymbol{\lambda}$-martingale or $\mathscr{Q}$-martingale), and $B \subseteq (Q \times \{0,1\}^*)^*$ (resp. $(\{0,1\}^*)^*$ or Q^*) a prefix set. Then,*

$$d(\lambda)\mu(\lambda) = d(\lambda) \geq \sum_{w \in B} d(w)\mu(w)$$

Proof.[*] If $d(\lambda) = 0$, this is immediate. Assume $d(\lambda) > 0$. Note that μ is a probability measure on $(Q \times \{0,1\}^*)^{\infty}$ because it satisfies the following conditions:

1. $\mu : (Q \times \{0,1\}^*)^* \to [0,1]$
2. $\mu(\lambda) = 1$.
3. If $|w| = n$ and $w = (q_0, u_0)...(q_{n-1}, u_{n-1})$ then,

$$\begin{aligned} \sum_{q \in Q} \mu(w(q, \lambda)) &= \sum_{q \in Q} \sigma(q_0)\Pi_{i=0}^{n-2}(q_i, q_{i+1})\Pi_{i=0}^{n-1}2^{-|u_i|}p(q_{n-1}, q) \\ &= \mu(w) \end{aligned} \tag{5.5.4}$$

4. If $|w| = n, u \in \{0,1\}^*$ and $w = (q_0, u_0)...(q_{n-2}, u_{n-2})(q_{n-1}, u)$ then,

$$\begin{aligned} \sum_{b \in \{0,1\}^*} \mu(wb) &= \sum_{b \in \{0,1\}^*} \sigma(q_0)(\Pi_{i=0}^{n-2}(q_i, q_{i+1}))(\Pi_{i=0}^{n-1}2^{-|u_i|})2^{-|ub|} \\ &= \mu(w) \end{aligned} \tag{5.5.5}$$

where wb is shorthand for $(q_0, u_0)...(q_{n-2}, u_{n-2})(q_{n-1}, ub)$.

Define $\pi : (Q \times \{0,1\}^*)^* \to [0,1]$ by

$$\sigma(w) = \frac{d(w)\mu(w)}{d(\lambda)}$$

[*]Kraft inequalities corresponding to $\boldsymbol{\lambda}$-martingales and $\mathscr{Q}$-martingales have nearly identical proofs and we omit these.

It is straightforward to show that this is a probability measure on $(Q \times \{0,1\}^*)^\infty$. Write

$$d(w) = d(\lambda)\frac{\pi(w)}{\mu(w)}$$

where π is a "strategy" and μ is the "environment".

Then, choose $\omega \in (Q \times \{0,1\}^*)^\infty$ according to π and let E be the event that $\exists w \in (Q \times \{0,1\}^*)^*)$ such that $w \sqsubseteq \omega$ for some $w \in B$ in this experiment. Then,

$$\begin{aligned} 1 &\geq Pr(E) \\ &= \sum_{w \in B} \pi(w) \\ &= \frac{1}{d(\lambda)} \sum_{w \in B} d(w)\mu(w) \end{aligned} \tag{5.5.6}$$

So,

$$d(\lambda) \geq \sum_{w \in B} d(w)\mu(w)$$

□

Lemma 4. *Let $d_0, d_1, d_2, d_3, ...$ be a sequence of C-martingales (resp. $\boldsymbol{\lambda}$-martingales or $\mathcal{Q}$-martingales) such that*

$$\sum_{n=0}^{\infty} d_n(\lambda) < \infty$$

Then, the function $d : (Q \times \{0,1\}^*)^* \to [0, \infty)$ (resp. $(\{0,1\}^*)^*$ or Q^*) defined by: $\forall w$,

$$d(w) = \sum_{n=0}^{\infty} d_n(w)$$

is a C-martingale (resp. $\boldsymbol{\lambda}$-martingale or $\mathcal{Q}$-martingale).

Proof. Let $d_0, d_1, ...$ and d be as given.

$\forall w \in (Q \times \{0,1\}^*)^*, q \in Q, u \in \{0,1\}^*$

$$\begin{aligned}
&\sum_{b\in\{0,1\}} d(w(q,ub))\mu(w(q,ub)) \\
&= \mu(w(q,u)) \sum_{b\in\{0,1\}} (\sum_{n=0}^{\infty} d_n(w(q,ub))) \\
&= \mu(w(q,u)) \sum_{n=0}^{\infty} \sum_{b\in\{0,1\}} d_n(w(q,ub)) \\
&= \mu(w(q,u)) \sum_{n=0}^{\infty} d_n(w(q,u)) \\
&= \mu(w(q,u))d(w(q,u))
\end{aligned} \tag{5.5.7}$$

and

$$\begin{aligned}
\sum_{q\in Q} d(w(q,\lambda))\mu(w(q,\lambda)) &= \mu(w) \sum_{q\in Q} (\sum_{n=0}^{\infty} d_n(w(q,\lambda))) \\
&= \mu(w) \sum_{n=0}^{\infty} \sum_{q\in Q} d_n(w(q,\lambda)) \\
&= \mu(w) \sum_{n=0}^{\infty} d_n(w) \\
&= \mu(w)d(w)
\end{aligned} \tag{5.5.8}$$

Since $d(\lambda)$ is finite and the martingale conditions hold, it follows by simple induction that $\forall w, d(w)$ is also finite. Thus, d is a C-martingale. □

Theorem 5.5.1. *For every CTMC C and every set $X \subseteq \Omega[C]$, the following two conditions are equivalent.*

(1) $\mu(X) = 0$

(2) There is a C-martingale d such that $X \subseteq S^{\infty}[d]$.

Proof. Suppose $\mu(X) = 0$. We wish to show that there exists a C-martingale, d, such that $X \subseteq S^{\infty}[d]$.

Assume the hypothesis. Then, $\forall k \in \mathbb{N} \exists C_k \subseteq (Q \times \{0,1\}^*)^*$ such that

$$X \subseteq \bigcup_{w\in C_k} \Omega_w$$

and

$$\sum_{w \in C_k} \mu(\Omega_w) \leq 2^{-k}$$

Let $k \in \mathbb{N}$. Suppose there exists $C_k \subseteq (Q \times \{0,1\}^*)^*$ satisfying the above conditions. Then, there exists $k \in \mathbb{N}$ and $g : \mathbb{N} \times \mathbb{N} \to (Q \times \{0,1\}^*)^* \cup \{\emptyset\}$, with the property that

- $X \subseteq \bigcup_{n=0}^{\infty} \Omega_{g(k,n)}$
- $\sum_{n=0}^{\infty} \mu(g(k,n)) \leq 2^{-k}$

We must define a martingale which succeeds on every $\boldsymbol{\tau} \in X \cap \Omega_{g(k,n)}$. Let $\boldsymbol{\tau} \in X \cap \Omega_{g(k,n)}$. Define the function $d_k : (Q \times \{0,1\}^*)^* \to [0, \infty)$ by

$$d_k(\lambda) = 2^{-k}$$

$$d_k(w) = \frac{\sum_{n=0}^{\infty} \mu(g(k,n) \wedge w)}{\mu(w)}$$

and where

$$\wedge : (Q \times \{0,1\}^*)^* \times (Q \times \{0,1\}^*)^* \to (Q \times \{0,1\}^*)^* \cup \emptyset$$

is defined by

$$x \wedge y = \begin{cases} x & \text{if } y \sqsubseteq x \\ y & \text{if } x \sqsubseteq y \\ \emptyset & otherwise \end{cases} \tag{5.5.9}$$

d_k is a C-martingale if it satisfies the conditions:

$$1A. \forall w \in (Q \times \{0,1\}^*)^*,$$

$$d(w)\mu(w) = \sum_{q \in Q} d(w(q,\lambda))\mu(w(q,\lambda))$$

$$2A. \forall w \in (Q \times \{0,1\}^*)^*, q \in Q, u \in \{0,1\}^*,$$

$$d(w(q,u))\mu(w(q,u)) = \sum_{b\in\{0,1\}} d(w(q,ub))\mu(w(q,ub))$$

Let $k \in \mathbb{N}, q \in Q, u \in \{0,1\}, w \in (Q \times \{0,1\}^*)^*$. To see that (1A) is satisfied,

$$\begin{aligned}
&\sum_{q\in Q} d_k(w(q,\lambda))\mu(w(q,\lambda)) \\
&= \sum_{q\in Q} [\frac{\sum_{n=0}^{\infty} \mu(g(k,n) \wedge (w(q,\lambda)))}{\mu(w(q,\lambda))}]\mu(w(q,\lambda)) \\
&= \sum_{q\in Q} \sum_{n=0}^{\infty} \mu(g(k,n) \wedge w(q,\lambda)) \\
&= \sum_{n=0}^{\infty} \sum_{q\in Q} \mu(g(k,n) \wedge w(q,\lambda)) \\
&= \sum_{n=0}^{\infty} \mu(g(k,n) \wedge w) \\
&= d_k(w)\mu(w)
\end{aligned} \tag{5.5.10}$$

To see that (2A) is satisfied,

$$\begin{aligned}
&\sum_{b\in\{0,1\}} d_k(w(q,ub))\mu(w(q,ub)) \\
&= \sum_{b\in\{0,1\}} [\frac{\sum_{n=0}^{\infty} \mu(g(k,n) \wedge (w(q,ub)))}{\mu(w(q,ub))}]\mu(w(q,ub)) \\
&= \sum_{b\in\{0,1\}} \sum_{n=0}^{\infty} \mu(g(k,n) \wedge w(q,ub)) \\
&= \sum_{n=0}^{\infty} \sum_{b\in\{0,1\}} \mu(g(k,n) \wedge w(q,ub)) \\
&= \sum_{n=0}^{\infty} \mu(g(k,n) \wedge w(q,u)) \\
&= d_k(w(q,u))\mu(w(q,u))
\end{aligned} \tag{5.5.11}$$

Hence, $\forall k \in \mathbb{N}, d_k$ is a C-martingale.

Define the unitary success set of a martingale d to be

$$S^1[d] = \{\boldsymbol{\tau} \in (Q \times (0,\infty))^\infty | (\exists w \sqsubseteq \boldsymbol{\tau}) d(w) \geq 1\}$$

Let $n \in \mathbb{N}, \boldsymbol{\tau} \in \Omega_{g(k,n)}$. Then, $g(k,n) \sqsubseteq \boldsymbol{\tau}$ and

$$d_k(g(k,n)) \geq \frac{\mu(g(k,n) \wedge g(k,n))}{\mu(g(k,n))} = 1$$

Thus, $\boldsymbol{\tau} \in S^1[d_k]$, and $\Omega_{g(k,n)} \subseteq S^1[d_k]$.

For each $k \in \mathbb{N}$, define $\hat{d}_k : (Q \times \{0,1\}^*)^* \to [0,\infty)$ by

$$\hat{d}_k(\lambda) = d_k(\lambda)$$

$$\hat{d}_k(wa) = \begin{cases} d_k(wa) & \text{if } \hat{d}_k(w) < 1 \\ \hat{d}_k(w) & \text{if } \hat{d}_k(w) \geq 1 \end{cases} \tag{5.5.12}$$

$\hat{d}_k$ is a C-martingale. Define $\hat{d} : (Q \times \{0,1\}^*)^* \to [0,\infty)$ by

$$\hat{d}(w) = \sum_{k=0}^{\infty} \hat{d}_k(w)$$

$\hat{d}$ is a C-martingale with the property that $X \subseteq S^\infty[d]$. To see this, let $\boldsymbol{\tau} \in X, \alpha \in \mathbb{Z}^+$. It suffices to show that there exists $x \sqsubseteq \boldsymbol{\tau}, \hat{d}(x) \geq \alpha$.

Since $\boldsymbol{\tau} \in X, \forall k \in \mathbb{N}, \boldsymbol{\tau} \in S^1[d_k]$. Then $\forall w \sqsubseteq \boldsymbol{\tau}, 0 \leq k < \alpha, \hat{d}_k(w) \geq 1$. Then, $\forall w \sqsubseteq \boldsymbol{\tau}$,

$$\hat{d}(w) \geq \sum_{k=0}^{\alpha-1} d_k(w) \geq \alpha$$

so there must exist $x \sqsubseteq \boldsymbol{\tau}$ such that $\hat{d}(x) \geq \alpha$. Thus, one direction is proven.

Now let $C = (Q, \boldsymbol{\lambda}, \pi)$ be a CTMC. Let $X \subseteq \Omega[C]$. Suppose there exists a C-martingale, d such that $X \subseteq S^\infty[d]$. Then, $\forall \boldsymbol{\tau} \in X, \alpha > 0, \exists w \in (Q \times \{0,1\}^*)^*$ such that $w \sqsubseteq \boldsymbol{\tau}$ and $d(w) > \alpha$. We wish to show that $\mu(X) = 0$.

We will show that there exists $g : \mathbb{N} \times \mathbb{N} \to (Q \times \{0,1\}^*)^* \cup \{\emptyset\}$, with the property that

(1B) $X \subseteq \bigcup_{n=0}^{\infty} \Omega_{g(k,n)}$

(2B) $\sum_{n=0}^{\infty} \mu(\Omega_{g(k,n)}) \leq 2^{-k}$

For each $k \in \mathbb{N}$, define

$$A_k = \{w \in (Q \times \{0,1\}^*)^* | d(w) \geq 2^k d(\lambda)\}$$

and

$$B_k = \{w \in A_k | \forall v \sqsubseteq w, v \notin A_k\}$$

B_k is thus the set of all partial specifications "by which" d has accumulated 2^k value for the first time along the unique path that is each $w \in B_k$.

For all $k \in \mathbb{N}$, define $B_k(i)$ to be the i-th element of B_k in standard enumeration of strings and define the function $g : \mathbb{N} \times \mathbb{N} \to (Q \times \{0,1\}^*)^* \cup \emptyset$ by

$$g(k,n) = \begin{cases} B_k(n) & \text{if } |B_k| \geq n \\ \emptyset & \text{otherwise} \end{cases} \tag{5.5.13}$$

To see that (1B) is satisfied, let $k \in \mathbb{N}, \boldsymbol{\tau} \in X$, and let d_k be defined as in the previous section. Since $\boldsymbol{\tau} \in S^{\infty}[d_k]$, $\exists w \in B_k$ s.t. $w \sqsubseteq \boldsymbol{\tau}$. Then, $\exists n \in \mathbb{N}$ s.t. $g(k,n) \sqsubseteq \boldsymbol{\tau}$, whence

$$\boldsymbol{\tau} \in \bigcup_{n=0}^{\infty} \Omega_{g(k,n)}$$

and we have that

$$X \subseteq \bigcup_{n=0}^{\infty} \Omega_{g(k,n)}$$

To see that (2B) is satisfied, by Lemma 3,

$$\begin{aligned} d(\lambda) &\geq \sum_{w \in B_k} d(w)\mu(w) \\ &\geq 2^k d(\lambda) \sum_{w \in B_k} \mu(w) \\ &= 2^k d(\lambda) \sum_{n=0}^{\infty} \mu(g(k,n)) \end{aligned} \tag{5.5.14}$$

and

$$\sum_{n=0}^{\infty} \mu(g(k,n)) \leq 2^{-k}$$

Thus, $\mu(X) = 0$. □

Theorem 5.5.2. *For every CTMC C and every set $X \subseteq \Omega[C]$, the following two conditions are equivalent.*

(1) $\mu_{\text{constr}}(X) = 0$.

(2) There is a lower semi-computable C-martingale d such that $X \subseteq S^{\infty}[d]$.

Proof. This proof follows the structure of the above proof with some adjustments:

Assume $\mu_{\text{constr}}(X) = 0$. Then, $\exists g : \mathbb{N} \times \mathbb{N} \to (Q \times \{0,1\}^*)^*$ such that g is computable and

$$X \subseteq \bigcup_{n=0}^{\infty} g(k,n) \text{ and } \sum_{n=0}^{\infty} \mu(g(k,n)) \leq 2^{-k}$$

Consider the same construction as before, and fix some $k \in \mathbb{N}$. Let M_k be the machine enumerating $g(k,0), g(k,1), \ldots$. To show that $d = \sum d_k$ is lower semicomputable, define

$$\hat{d}(w,t) = \sum_{k=0}^{\infty} \frac{\mu((C_k)_t \cap C_w)}{\mu(C_w)}$$

where

$$(C_k)_t = \bigcup_{n=0}^{t} g(k,n) \text{ and } \lim_{t\to\infty} (C_k)_t = C_k$$

for each $k \in \mathbb{N}$.

Clearly, $\hat{d}(w,t) \leq \hat{d}(w,t+1) < d(w)$ and $\lim_{t\to\infty} \hat{d}(w,t) = d(w)$ for all w, t.

Assume instead that there exists a constructive martingale d, with $X \subseteq S^{\infty}[d]$ and a function $\hat{d}$ testifying to the lower semi-computability of d. We wish to show that for each $k \in \mathbb{N}$, the set A_k is computably enumerable.

Define an enumerator M_k: For each $(w,t) \in (Q \times \{0,1\}^*)^* \times \mathbb{N}$, dovetailing, compute $\hat{d}(w,t)$. If $\hat{d}(w,t) \geq 2^k d(\lambda)$, output w.

M_k enumerates

$$A_k = \{w \in (Q \times \{0,1\}^*)^* | d(w) \geq 2^k d(\lambda)\}$$

A prefix set $B_k \subseteq A_k$ can be enumerated by running the enumerator for A_k and not enumerating any element for which a prefix has been printed or which would prefix and already printed element. The resulting function

$$g(k,n) = \begin{cases} B_k(n) & \text{if } |B_k| \geq n \\ \emptyset & \text{otherwise} \end{cases} \tag{5.5.15}$$

produces a constructive null cover of X. □

Much like the classical setting, we call a trajectory $\boldsymbol{\tau}$ *Martin-Löf random* if $\{\boldsymbol{\tau}\}$ is not of constructive measure 0.

We use the shorthand wb, where $w = (q_0, u_0), ..., (q_k, u_k)$, and $b \in \{0, 1\}$, to denote $(q_0, u_0)...(q_k, u_k b)$ and wq, where $q \in Q$, to denote $(q_0, u_0)...(q_k, u_k), (q, \lambda)$.

Lemma 5.5.3. *Let $C = (Q, \boldsymbol{\lambda}, \pi)$ be a CTMC. Then, the following hold for any resource bound $\Delta \in \{\text{constr}, \text{comp}, \text{all}\}$:*

1. If $S' \subseteq Q^\infty$ and d is a Δ-computable C-martingale which succeeds on $S = \bigcup_{\boldsymbol{q} \in S'} \bigcup_{\boldsymbol{r} \in \mathbb{R}^\infty} (\boldsymbol{q}, \boldsymbol{r})$ then there is a Δ-computable $\mathcal{2}$-martingale d' which succeeds on S'.

2. If $\boldsymbol{\kappa} = (\kappa_{q_1}, \kappa_{q_2}, ...)$ is a rate sequence corresponding to Δ-computable state sequence $q_0, q_1, q_2, ... \in Q^\infty$, $T' \subseteq \boldsymbol{D}_{\boldsymbol{\kappa}}$, and d is a Δ-computable C-martingale which succeeds on $T = \bigcup_{\boldsymbol{q} \in Q^{\leq \infty}} \bigcup_{\boldsymbol{r} \in T'} (\boldsymbol{q}, \boldsymbol{r})$, then there is a Δ-computable $\boldsymbol{\kappa}$-martingale which succeeds on T'.

Proof. 1. Assume the hypothesis. Define d' by

$$d'(q_0, ..., q_k) = d((q_0, \lambda), (q_1, \lambda), ..., (q_k, \lambda)).$$

To see that d succeeds on every $\boldsymbol{q} \in S'$, for each $i \in \mathbb{N}$ define $t_i \in \{0, 1\}^\infty$ by

$$t_i[j] = 1 \text{ iff}$$

$$d((q_0, \lambda)(q_1, \lambda)...(q_i, t_i[0...j-1]1)) \leq d((q_0, \lambda)(q_1, \lambda)...(q_i, t_i[0...j-1]0))$$

and $t_i[j] = 0$ otherwise. In other words, t_i is an 'artificially' constructed 'minimal-payoff' time for q_i. Lastly, note that for all $u_i \sqsubseteq t_i$ and $u_0...u_{i-1}$,

$$d((q_0, u_0)(q_1, u_1)...(q_i, u_i)) \leq d((q_0, u_0)(q_1, u_1)...(q_i, \lambda))$$

Since d succeeds on S, d' succeeds on S'. Since d is Δ-computable, so is d'.

2. Assume the hypothesis. If $\Delta = \text{constr}$, let $\epsilon_i = 2^{-i}$, otherwise let $\epsilon = 0$. Since d succeeds on $T = \bigcup_{\boldsymbol{q} \in Q^{\leq \infty}} \bigcup_{\boldsymbol{r} \in T'} (\boldsymbol{q}, \boldsymbol{r})$, there exists a $\boldsymbol{q} = (q_0, q_1, ...)$ such that: for all $i \in \mathbb{N}$, $u_0, ...u_{i-1}$,

$$q_i = q \text{ iff}$$

$$d((q_0, u_0)(q_1, u_1)...(q_{i-1}, u_{i-1}), (q, \lambda)) \leq d((q_0, u_0)(q_1, u_1)...(q_{i-1}, u_{i-1}), (q', \lambda)) + \epsilon_i$$

for all $q' \in Q$ if such a q exists (otherwise, let $q_i = \min_{i \in \mathbb{N}}\{s_i \mid d(...(s_i, u_i) < d(...(s_{i-1}, u_{i-1}) + \epsilon\})$. d succeeds on $(\boldsymbol{q}, \boldsymbol{t})$ for every $\boldsymbol{t} \in T'$.

In other words, $\boldsymbol{q}$ is an 'artificially' constructed 'minimal-payoff' state sequence. Now, define d' by

$$d'(u_0, ..., u_k) = d((q_0, u_0), ..., (q_k, u_k)).$$

It is intuitively useful to conceive of d' as gambling on times according to $\boldsymbol{\kappa}$ and making only finite money by gambling on the states of $\boldsymbol{q}$. (If $\Delta = \text{constr}$, then the particular values of d can be estimated with increasing accuracy (2^{-i} as $i \to \infty$).

Since d succeeds on T, it follows that d' succeeds on T'.

□

We now give a proof of Theorem 1. *Proof.* Let $C = (Q, \boldsymbol{\lambda}, \pi)$ be a CTMC. Let d' be a $\mathscr{Q}$-martingale which succeeds on some set $S' \subseteq Q$. Define a C-martingale d for all $q \in Q, u \in \{0,1\}^*$, and partial specifications $w = (q_0, u_0), (q_1, u_1), ...(q_k, u_k)$ by

$$d(w(q, u)) = d'(q_0, ...q_k, q).$$

Then, d succeeds on the set $S = \{\tau = (\boldsymbol{q}, \boldsymbol{r}) \in (Q \times \{0,1\}^\infty)^\infty \mid \boldsymbol{q} \in S'\}$. Then, $\mu_C(S) = 0$, by Theorem 5.5.1. Then,

$$\mu_C(\bigcup_{\boldsymbol{q} \in S'} \bigcup_{\boldsymbol{r} \in \mathbb{R}^\infty} (\boldsymbol{q}, \boldsymbol{r})) = 0.$$

Thus, $\mu_Q(S') = 0$.

Now assume instead that $\mu_Q(S') = 0$. Then,

$$\mu_C(\bigcup_{\boldsymbol{q}\in S'}\bigcup_{\boldsymbol{r}\in\mathbb{R}^\infty}(\boldsymbol{q},\boldsymbol{r})) = 0.$$

Then, by Theorem 5.5.1 there is a C-martingale d which succeeds on S. Then by Lemma 5.5.3, there is a martingale d' which succeeds on S'. □

We now give a proof of Theorem 2.

Proof. Let $C = (Q, \boldsymbol{\lambda}, \pi)$ be a CTMC and $\boldsymbol{\kappa} = (\kappa_{q_1}, \kappa_{q_2}, ...)$ a rate sequence corresponding to state sequence $\boldsymbol{q} = (q_1, q_2, ...) \in Q^\infty$.

Let d' be a $\boldsymbol{\kappa}$-martingale which succeeds on some set $T' \subseteq \boldsymbol{D}_{\boldsymbol{\kappa}}$. Then, define a C-martingale d as follows for all $\boldsymbol{s}$, and $u_0, ...u_k$:

$$d(\boldsymbol{s} \upharpoonright k, (u_0, ...u_k)) = d'(u_0, ..., u_k)$$

It is straightforward to verify that d is a martingale and d succeeds on $T = \bigcup_{\boldsymbol{r}\in T'}(\boldsymbol{q},\boldsymbol{r})$, which has $\mu_C(T) = 0$, whence $\mu_{\boldsymbol{\kappa}}(T') = 0$ as well.

Now assume instead that $T' \subseteq \boldsymbol{D}_{\boldsymbol{\kappa}}$ has $\mu_{\boldsymbol{\kappa}}(T') = 0$. Then $T = \bigcup_{\boldsymbol{q}\in Q^{\leq\infty}}\bigcup_{\boldsymbol{r}\in T'}(\boldsymbol{q},\boldsymbol{r})$ has $\mu_C(T) = 0$. Then, by Theorem 5.5.1 there exists a C-martingale d which succeeds on T. By Lemma 5.5.3 there exists a $\boldsymbol{\kappa}$-martingale which succeeds on T'.

□

Proofs of constructive versions of theorems 1 and 2 are identical to the above proofs, resource bounds as needed.

Lemma 5.5.4. *Let C be a CTMC and $\boldsymbol{\tau} = (q_0, t_0)(q_1, t_1)....\in \Omega[C]$ be random. Then, the subsequence consisting of all states in $\boldsymbol{\tau}$, $\boldsymbol{q} = q_0, q_1, q_2, ... \in Q^\infty$ is random with respect to $(\mathscr{Q}, \sigma)$.*

Proof. Let $\boldsymbol{\tau}, \boldsymbol{q}$ be as described. Suppose there exists a lower semicomputable $(\mathscr{Q}, \sigma)$-martingale $d : Q^* \to [0, \infty)$ which succeeds on $\boldsymbol{q}$ (that is, $\boldsymbol{q}$ is not random).

Define the C-martingale $\hat{d} : (Q \times \{0, 1\}^*)^* \to [0, \infty)$ as follows:

If $q \in Q$

$$\hat{d}(wq) = d(q_0, ..., q_{n-1}, q)$$

If $b \in \{0, 1\}$

$$\hat{d}(wb) = \hat{d}(w)$$

That is, $\hat{d}$ only bets on states (and bets on them according to d's strategy), while hedging its bets on times. To see that $\hat{d}$ is in fact a C-martingale:

$\forall w \in (Q \times \{0,1\}^*)^*, q \in Q, u \in \{0,1\}^*$

$$\begin{aligned} \sum_{b \in \{0,1\}} \hat{d}(w(q,ub))\mu(w(q,ub)) &= \hat{d}(w(q,u)) \sum_{b \in \{0,1\}} \mu(w(q,ub)) \\ &= \hat{d}(w(q,u))\mu(w(q,u)) \end{aligned} \tag{5.5.16}$$

and $\forall w \in (Q \times \{0,1\}^*)^*, |w| = n$

$$\begin{aligned} \sum_{q \in Q} \hat{d}(w(q,\lambda))\mu(w(q,\lambda)) &= \sum_{q \in Q} d(q_0, ..., q_{n-1}, q)\mu(w(q,\lambda)) \\ &= d(q_0, ..., q_{n-1}) \sum_{q \in Q} \mu(w(q,\lambda)) \\ &= \hat{d}(w) \sum_{q \in Q} \mu(w(q,\lambda)) \\ &= \hat{d}(w)\mu(w) \end{aligned} \tag{5.5.17}$$

To see that $\hat{d}$ succeeds on $\boldsymbol{\tau}$ let $\alpha > 0$. Since d succeeds on $\boldsymbol{q}$, $\exists n \in \mathbb{N}$ and $w_n \sqsubseteq \boldsymbol{q}$ such that $d(w_n) > \alpha$. Then, since $\hat{d}$ does not bet on sojourn times and bets on states according to d,

$$\hat{d}((q_0, u_0)(q_1, u_1)...(q_{n-1}, u_{n-1})) > \alpha$$

To see that $\hat{d}$ is lower semicomputable, let $d' : Q^* \times \mathbb{N} \to \mathbb{Q}$ be a function testifying to the fact that d is lower semicomputable. Define $\hat{d}'$ as $\hat{d}$ is defined above, replacing instances of d with instances of d'. □

Lemma 5.5.5. *Let $\boldsymbol{\tau} = (q_0, t_0)(q_1, t_1)..... \in \Omega[C]$ where C is some CTMC. Suppose $\exists m \in \mathbb{N}$ such that t_m is not random. Then, $\boldsymbol{\tau}$ is not random.*

Proof. Assume the hypothesis and let d be a lower semi-computable martingale which succeeds on some t_m. For every $n \in \mathbb{N}$ define a C-martingale $\hat{d}_n$ which

1.Doesn't bet on states

2.Bets according to d on only the nth sojourn time $t_n (n = 0...\infty)$.

$$\hat{d}_n(\lambda) = 2^{-n},$$

$$\hat{d}_n(w(q,\lambda)) = \hat{d}_n(w)$$

If $|w| = n, w = (q_0, u_0)(q_1, u_1)...(q_{n-1}, u), u \in \{0,1\}^*, b \in \{0,1\}$

$$\hat{d}_n(w[0...n-2](q_{n-1}, ub)) = d(ub)$$

If $|w| = k \neq n, w = (q_0, u_0)(q_1, u_1)...(q_{k-1}, u), u \in \{0,1\}^*, b \in \{0,1\}$

$$\hat{d}_n(w[0...k-2](q_{k-1}, ub)) = \hat{d}_n(w[0...k-2](q_{k-1}, u))$$

Let $n \in \mathbb{N}$. We must prove $\hat{d}_n$ is indeed a martingale.

If $q \in Q$,

$$\begin{aligned} \sum_{q \in Q} \hat{d}_n(w(q,\lambda))\mu(w(q,\lambda)) &= \sum_{q \in Q} \hat{d}_n(w)\mu(w(q,\lambda)) \\ &= \hat{d}_n(w) \sum_{q \in Q} \mu(w(q,\lambda)) \\ &= \hat{d}_n(w)\mu(w) \end{aligned} \tag{5.5.18}$$

If $|w| = k \neq n$,

$$\begin{aligned} &\sum_{b \in \{0,1\}} \hat{d}_n(w[0...n-2](q,ub))\mu(w[0...n-2](q,ub)) \\ &= \sum_{b \in \{0,1\}} d(ub)\mu(w[0...n-2](q,ub)) \\ &= d(u)\mu(w[0...n-2](q,u)) \\ &= \hat{d}(w[0...n-2](q,u))\mu(w[0...n-2](q,u)) \end{aligned} \tag{5.5.19}$$

If $|w| = n$,

$$\begin{aligned}
&\sum_{b\in\{0,1\}} \hat{d}_n(w[0...n-2](q,ub))\mu(w[0...n-2](q,ub)) \\
&= \sum_{b\in\{0,1\}} d(ub)\mu(w[0...n-2](q,ub)) \\
&= d(u)\mu(w[0...n-2](q,u)) \\
&= \hat{d}(w[0...n-2](q,u))\mu(w[0...n-2](q,u))
\end{aligned} \tag{5.5.20}$$

Define $\hat{d}$ to be a C-martingale obtained by applying Lemma 4:

$$\hat{d} = \sum_{n=0}^{\infty} \hat{d}_n$$

$\hat{d}$ succeeds on $\boldsymbol{\tau}$.

Since d is lower semicomputable, let d' testify to this. Substituting d' in the above construction shows that $\hat{d}_n$ is lower semicomputable for all n, and thus that $\hat{d}$ is also lower semicomputable. Thus, $\boldsymbol{\tau}$ is not random. □

Lemma 5.5.6. *Let $\boldsymbol{\tau} \in \Omega[C]$ be a trajectory in a CTMC, C. If $\boldsymbol{\tau}$ is random, then all sojourn times $t_0, t_1, t_2, ...$ in $\boldsymbol{\tau}$ are independently random.*

Proof. Let $\boldsymbol{\tau} \in \Omega[C]$ and suppose there exists n such that $t_1, ..., t_n$ are not independently random. Then, there exists $d : \{0,1\}_n^* \to [0, \infty)$ (where $\{0,1\}_n^*$ denotes the set of all n-tuples of strings *of the same length*) such that $\forall w \in \{0,1\}_n^*$

$$d(w)\mu(w) = \sum_{a\in 0,1_n^1} d(wa)\mu(wa)$$

and

$$\limsup_{k\to\infty} d((t_1, ..., t_n) \upharpoonright k) = \infty,$$

where μ refers to the probability measure on $\{0,1\}_n^*$ defined by

$$\mu((w_1, ..., w_n)) = \Pi_{i=1}^n \mu_i(w_i)$$

and d is lower semicomputable.

Define the martingale $d : \{0,1\}^* \to [0,\infty)$ by

$$d(w) = d(w, t_2, ..., t_3).$$

It's clear that is a martingale which succeeds on t_0, from which it follows that $\boldsymbol{t}$ and thus also $\boldsymbol{\tau}$ cannot be random. □

Lemma 5.5.7. *There exists a rate sequence $\boldsymbol{\lambda}$ and a sequence $\boldsymbol{R} = (\boldsymbol{t_0}, \boldsymbol{t_1}, ...)$ of $\boldsymbol{\lambda}$-durations such that $\boldsymbol{t_0}, \boldsymbol{t_1}, ...$ are independently random but $\boldsymbol{R}$ is not random with respect to $\mu_{\boldsymbol{\lambda}}$.*

Proof. Let $\boldsymbol{\lambda}$ be a rate sequence and let $S_0, S_1, ...$ be a sequence of elements of $\{0,1\}^\infty$ representing times $t_0, t_1, ...$ each of which are random with respect to the rates $\lambda_0, \lambda_1, ...$ Then, the times (as binary sequences) in the $\boldsymbol{\lambda}$-duration sequence $(0S_0, 0S_1, 0S_2, ..)$ are not independently random since a lower-semicomputable $\boldsymbol{\lambda}$-martingale exists which can bet only on the first bit of each sequence and hedge on all other bits. □

5.5.3 Chemical reaction networks

Like the *rate-free* chemical reaction networks described in section 2, a *chemical reaction network* (*CRN*) is an ordered pair, $N = (S, R)$ where S is a finite set of species, Q is a countable set of states $q \in \mathbb{N}^S$ representing integer quantities of each species, and $R = \{\rho_1, \rho_2, ..., \rho_r\}$ is a finite set of reactions. However, the *reactions* in a CRN are triples $\rho = (r, p, k)$, where r and p are as before, and the nonnegative real number k is a *rate constant* governing, along with the reactants r and state q, the *rate* λ_q of reaction ρ at state q. As before, for ρ to be *applicable* in state q, it must be that $r \leq q$, otherwise the reaction cannot occur at q.

Theorem 5.5.8 (Non-Zeno property)**.** *Let C be a CRN. Then, if $\boldsymbol{\tau} = (q_0, t_0), (q_1, t_1), ... \in \Omega[C]$ is random and has bounded molecular counts, then $\boldsymbol{\tau}$ satisfies the non-Zeno property that*

$$\sum_{i=0}^{\infty} t_i = \infty.$$

Proof. Let C be a CRN and $\boldsymbol{\tau} \in \Omega[C]$ a trajectory with bounded molecular counts. Since $\boldsymbol{\tau}$ has bounded molecular counts, there exists a constant $M \in \mathbb{R}$ which is the maximum reaction rate along $\boldsymbol{\tau}$. Since $\boldsymbol{\tau}$ has the Zeno property, there must exist $i \in \mathbb{N}$ such that $\forall k \geq i, t_k \in J(\lambda_{q_i}, 0)$. Define a C-martingale which only bets on the first bit of each sojourn time $t_i, t_{i+1}, ...$ as follows:

$$d_i(\lambda) = 1 \tag{5.5.21}$$

$$d_i(w(q, \lambda)) = d_i(w) \tag{5.5.22}$$

$$d_i(w(q, ub)) = \begin{cases} (2d_i(w(q, u)) & \text{if } |w| = i, u = \lambda, \text{ and } b = 0 \\ d_i(w(q, u)) & \text{if } |w| < i \\ 0 & \text{if } |w| = i, \text{ and } b \neq 0 \end{cases} \tag{5.5.23}$$

Since i is a definite value, d_i does not begin to bet until it reaches the i-th sojourn time, and d_i bets only on the first bit of each sojourn time after the ith, d_i succeeds on $\boldsymbol{\tau}$. d_i is clearly lower semicomputable. Thus, $\boldsymbol{\tau}$ cannot be random. □

5.5.4 Kolmogorov complexity characterization

Random trajectories can also be characterized using *Kolmogorov complexity*. First, we briefly review this notion in the classical setting. We fix a universal *self-delimiting Turing machine* (see [35]), U. The Kolmogorov complexity, K, of a (finite) string x in $\{0,1\}^*$ is the length of a shortest program for a self-delimiting Turing machine which prints x. That is, $K : \{0,1\}^* \to \mathbb{N}$ is defined by

$$K(x) = \min\{|\pi| \mid U(\pi) = x \text{ and } \pi \in \{0,1\}^*\}.$$

When x is not a binary string, but some other finite object, $K(x)$ is defined from the above by routine coding.

Definition 5. *The profile of a cylinder Ω_w of a CTMC is*

$$\text{prof}(w) = (|u_1|, ..., |u_n|),$$

where $w = ((q_1, u_1), ..., (q_n, u_n))$.

Observation 5.5.9. *For each CTMC* C *and each profile* p,

$$\sum_{\{w:\text{prof}(w)=p\}} \mu_C(w) = 1.$$

The following two lemmas are analogous to standard results used in the Kolmogorov complexity characterization of algorithmically random sequences.

Lemma 5.5.10. *For every cylinder,* Ω_w *of a CTMC* C,

$$K(w) \leq l(w) + K(\text{prof}(w)) + O(1),$$

where $l(w) = \log \frac{1}{\mu_C(w)}$ *is the "self-information" of* w.

*Proof.*In the following proof, we let p range over all profiles, and assume there is some natural encoding (enumerating process) between natural numbers and profiles, and also between natural numbers and cylinders.

$$\begin{aligned}
\Omega &= \sum_p 2^{-K(p)}, \\
&= \sum_p \left(2^{-K(p)} \sum_{\text{prof}(w)=p} 2^{-l(w)} \right), \quad \text{note that the second summation is 1.} \\
&= \sum_w 2^{-(K(\text{prof}(w))+l(w))} < \infty.
\end{aligned}$$

Then, by the minimality of K and the coding relation between cylinders and natural numbers, we have

$$K(w) \leq l(w) + K(\text{prof}(w)) + O(1).$$

□

Lemma 5.5.11. *There is a constant* $c \in \mathbb{N}$ *such that, for every profile* p *of a CTMC* C *and every* $k \in \mathbb{N}$,

$$\mu_C\Bigg(\bigcup_{\substack{w \\ prof(w)=p \\ K(w)<l(w)+K(p)-k}} \Omega_w \Bigg) < 2^{c-k}.$$

Substituting $k + K(\mathrm{prof}(w))$ *for* k *here gives*

$$\mu_C\Big(\bigcup_{\substack{w \\ prof(w)=p \\ K(w)<l(w)-k}} \Omega_w\Big) < 2^{c-k-K(p)}.$$

Proof. We only need to note that

$$\sum_p \sum_{\mathrm{prof}(w)=p} 2^{-K(w)} = \sum_w 2^{-K(w)} < \infty.$$

Then by the minimality [7] of K, we have,

$$\begin{aligned}
2^{-K(p)+c} &\geq \sum_{\mathrm{prof}(w)=p} 2^{-K(w)} \\
&= \sum_{\mathrm{prof}(w)=p} \mu(w) \frac{1}{\mu(w)} 2^{-K(w)} \\
&= \sum_{\mathrm{prof}(w)=p} \mu(w) 2^{\log \frac{1}{\mu(w)}} 2^{-K(w)} \\
&= \sum_{\mathrm{prof}(w)=p} \mu(w) 2^{l(w)-K(w)} \\
&= \mathbb{E}_\mu[2^{l(w)-K(w)}]
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\mu\{w \mid K(w) < l(w) + K(\mathrm{prof}(w)) - k)\} \\
&= \mu\{w \mid l(w) - K(w) > k - K(\mathrm{prof}(w)))\} \\
&= \mu\left\{w \mid 2^{l(w)-K(w)} > 2^{k-K(\mathrm{prof}(w)))}\right\} \\
&< \frac{\mathbb{E}_\mu[2^{l(w)-K(w)}]}{2^{k-K(\mathrm{prof}(w)))}} \leq \frac{2^{-K(p)+c}}{2^{k-K(\mathrm{prof}(w)))}} = 2^{c-k}
\end{aligned}$$

The first inequality in the last row follows by the Markov inequality. □

With these lemmas, we can establish the Kolmogorov complexity characterization of randomness for trajectory objects, which is exactly analogous to a well-known characterization of the algorithmic randomness of sequences over finite alphabets [68, 48].

Theorem 5.5.12. *A trajectory $\boldsymbol{\tau}$ is Martin Löf random if and only if there exists $k \in \mathbb{N}$, such that for every $w \sqsubseteq \boldsymbol{\tau}$, $K(w) \geq l(w) - k$.*

Proof. "Only if": We prove the contrapositive. Suppose that for every k, there is at least one $w \sqsubseteq \boldsymbol{\tau}$, such that $K(w) < l(w) - k$. We let

$$U_k = \{w \mid K(w) < l(w) - k\}.$$

Note that w ranges over all cylinders in the above definition. Therefore, it it clear that $\boldsymbol{\tau}$ is covered by the U_k.

Next, we are going to estimate the measure of U_k. First we consider the p-slice of U_k, U_k^p, defined as:

$$U_k^p = \{w \mid \text{prof}(w) = p \text{ and } w \in U_k\}, \quad \text{where } p \text{ is a profile.}$$

Note that by Lemma 5.5.11, we have $\mu[U_k^p] < 2^{c-k-K(p)}$, therefore

$$\mu[U_k] = \sum_p \mu[U_k^p] \leq \sum_t 2^{c-k-K(p)} \leq 2^{c-k}\Omega \leq 2^{c-k}.$$

Also note that each U_k is recursively enumerable, and $\{V_k\}_k$, where $V_k = U_{c+k}$ is a Martin Löf test.

"If": Again by contrapositive: Assume $\boldsymbol{\tau}$ is not Martin Löf random, and let $\{U_k\}$ be a Martin Löf test. We construct the following (output, size-of-program) requirement pairs as follows:

$$\{(w, l(w) - k) \mid w \in U_{k^2}, k \geq 2\}$$

It can be checked this requirement satisfies Kraft's inequality, since the measure of the size-of-program is bounded from above by

$$\sum_{k \geq 2} 2^{-(k^2-k)} = 1/2^2 + 1/2^6 + 1/2^{12} \cdots < 1$$

Then by Levin's coding lemma [33, 34], this requirement can be fulfilled.

Note that $\boldsymbol{\tau}$ can be covered by U_{k^2}, and therefore for each $k \geq 2$ there are prefixes w of $\boldsymbol{\tau}$ for which $K(w) \leq l(w) - k < l(w) - (k-1)$.

That is, for every $k' = k - 1 > 0$, there is some $w \sqsubseteq \boldsymbol{\tau}$, such that $K(w) < l(w) - k'$, Hence $\boldsymbol{\tau}$ is not random in the Kolmogorov sense.

□

BIBLIOGRAPHY

[1] mathworld.wolfram.com, 2020. Retrieved July 24, 2020, from https://mathworld.wolfram.com/Machin-LikeFormulas.html.

[2] Kelty Allen, Laurent Bienvenu, and Theodore Slaman. On zeros of Martin-Löf random Brownian motion. *Journal of Logic and Analysis*, 6, 2015.

[3] Olivier Bournez, Pierre Fraigniaud, and Xavier Koegler. Computing with large populations using interactions. In *Proceedings of the 37th International Conference on Mathematical Foundations of Computer Science*, pages 234–246. Springer, 2012.

[4] Olivier Bournez, Daniel S. Graça, and Amaury Pouly. Polynomial time corresponds to solutions of polynomial ordinary differential equations of polynomial length: The general purpose analog computer and computable analysis are two efficiently equivalent models of computations. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming*, volume 55 of *Leibniz International Proceedings in Informatics*, pages 109:1–109:15. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016.

[5] Richard P Brent. Fast multiple-precision evaluation of elementary functions. *Journal of the ACM (JACM)*, 23(2):242–251, 1976.

[6] Vannevar Bush. The differential analyzer. A new machine for solving differential equations. *Journal of the Franklin Institute*, 212(4):447–488, 1931.

[7] Gregory J. Chaitin. Incompleteness theorems for random reals. *Advances in Applied Mathematics*, 8(2):119–146, 1987.

[8] Yuan-Jyue Chen, Neil Dalchau, Niranjan Srinivas, Andrew Phillips, Luca Cardelli, David Soloveichik, and Georg Seelig. Programmable chemical controllers made from DNA. *Nature Nanotechnology*, 8(10):755–762, 2013.

[9] Matthew Cook, David Soloveichik, Erik Winfree, and Jehoshua Bruck. Programmability of chemical reaction networks. In Anne Condon, David Harel, Joost N. Kok, Arto Salomaa, and Erik Winfree, editors, *Algorithmic Bioprocesses*, Natural Computing Series, pages 543–584. Springer, 2009.

[10] Wojciech Czerwinski, Slawomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. The reachability problem for Petri nets is not elementary. *arXiv preprint arXiv:1809.07115*, 2018.

[11] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic Randomness and Complexity*. Springer Science & Business Media, 2010.

[12] François Fages, Guillaume Le Guludec, Olivier Bournez, and Amaury Pouly. Strong Turing completeness of continuous chemical reaction networks and compilation of mixed analog-digital programs". In *"Proceedings of the 15th International Conference on Computational Methods in Systems Biology*, pages 108–127. Springer International Publishing, 2017.

[13] Francois Fages, Guillaume Le Guludec, Olivier Bournez, and Amaury Pouly. Turing completeness of elementary reaction systems under the differential semantics and compilation of mixed analog-digital programs. 2017.

[14] Patrick C Fischer, Albert R Meyer, and Arnold L Rosenberg. Time-restricted sequence generation. *Journal of Computer and System Sciences*, 4(1):50–73, 1970.

[15] Willem L. Fouché. Fractals generated by algorithmically random Brownian motion. In *Conference on Computability in Europe*, pages 208–217. Springer, 2009.

[16] Willem L. Fouché. Kolmogorov complexity and the geometry of Brownian motion. *Mathematical Structures in Computer Science*, 25(7):1590–1606, 2015.

[17] Mrinalkanti Ghosh and Satyadev Nandakumar. Predictive complexity and generalized entropy rate of stationary ergodic processes. In *International Conference on Algorithmic Learning Theory*, pages 365–379. Springer, 2012.

[18] Timothy Gowers, editor. *The Princeton Companion to Mathematics*. Princeton University Press, 2008.

[19] Daniel S Graça and Amaury Pouly. Computational complexity of solving polynomial differential equations over unbounded domains. *Theoretical Computer Science*, 626(2):67–82, 2016.

[20] Daniel Silva Graça and José Félix Costa. Analog computers and recursive functions over the reals. *Journal of Complexity*, 19(5):644–664, 2003.

[21] Daniel Silva Graça. Some recent developments on Shannon's general purpose analog computer. *Mathematical Logic Quarterly*, 50(4-5), 2004.

[22] Yuri Gurevich and Saharon Shelah. Nearly linear time. In *International Symposium on Logical Foundations of Computer Science*, pages 108–118. Springer, 1989.

[23] Vera Hárs and János Tóth. On the inverse problem of reaction kinetics. *Qualitative Theory of Differential Equations*, 30:363–379, 1981.

[24] Juris Hartmanis and Richard E Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[25] Morris W Hirsch and Stephen Smale. *Differential Equations, Dynamical Systems, and Linear Algebra*. Academic Press, 1974.

[26] Xiang Huang, Titus H. Klinge, James I. Lathrop, Xiaoyuan Li, and Jack H. Lutz. Real-time computability of real numbers by chemical reaction networks. *Natural Computing*, 18(1):63–73, 2019.

[27] Ekatharine A Karatsuba. On the computation of the euler constant γ. *Numerical Algorithms*, 24(1-2):83–97, 2000.

[28] Bjørn Kjos-Hanssen and Anil Nerode. Effective dimension of points visited by Brownian motion. *Theoretical Computer Science*, 410(4-5):347–354, 2009.

[29] Ker-I Ko. *Complexity Theory of Real Functions*. Birkhäuser, 1991.

[30] Xavier Koegler. *Population Protocols, Games, and Large Populations*. PhD thesis, Université Paris Diderot - Paris 7, 2012.

[31] Thomas G. Kurtz. The relationship between stochastic and deterministic models for chemical reactions. *The Journal of Chemical Physics*, 57(7):2976–2978, 1972.

[32] Jérôme Leroux and Sylvain Schmitz. Demystifying reachability in vector addition systems. In *LICS 2015*, pages 56–67. IEEE, 2015.

[33] Leonid A. Levin. On the notion of a random sequence. *Soviet Mathematics Doklady*, 14(5), 1973.

[34] Leonid A. Levin. Laws of information conservation (nongrowth) and aspects of the foundation of probability theory. *Problemy Peredachi Informatsii*, 10(3):30–35, 1974.

[35] Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer Science & Business Media, 2013.

[36] Leonard Lipshitz and Lee A Rubel. A differentially algebraic replacement theorem, and analog computability. *Proceedings of the American Mathematical Society*, 99(2):367–372, 1987.

[37] Richard J. Lipton. Why the Hartmanis-Stearns conjecture is still open, 2012. Blog post. Retrieved February 3, 2017, from https://rjlipton.wordpress.com/2012/06/15/why-the-hartmanis-stearns-conjecture-is-still-open/.

[38] Neil Lutz. Fractal intersections and products via algorithmic dimension. In *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017)*, pages 58:1–58:12. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[39] Neil Lutz and Donald M. Stull. Bounding the dimension of points on a line. In *International Conference on Theory and Applications of Models of Computation*, pages 425–439. Springer, 2017.

[40] A. M. Lyapunov. The general problem of the stability of motion, 1992. English translation by A.T. Fuller of Lyapunov's 1892 Ph.D. thesis at Moscow University.

[41] Marcelo O. Magnasco. Chemical kinetics is Turing universal. *Physical Review Letters*, 78(6):1190–1193, 1997.

[42] D. Marker. *Model Theory : An Introduction*. Graduate Texts in Mathematics. Springer New York, 2002.

[43] Satyadev Nandakumar. An effective ergodic theorem and some applications. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 39–44. ACM, 2008.

[44] André Nies. *Computability and Randomness*, volume 51. OUP Oxford, 2009.

[45] Amaury Pouly. *Continuous models of computation: from computability to complexity*. PhD thesis, Université de Caen Basse-Normandie, 2015.

[46] Marion B Pour-El and Jonathan I Richards. Abstract computability and its relations to the general purpose analog computer. *Transactions of the American Mathematical Society*, 199:1–28, 1974.

[47] Claus-Peter Schnorr. A unified approach to the definition of random sequences. *Mathematical Systems Theory*, 5(3):246–258, 1971.

[48] Claus-Peter Schnorr. A survey of the theory of random sequences. In *Basic Problems in Methodology and Linguistics*, pages 193–211. Springer, 1977.

[49] Claus-Peter Schnorr and Hermann Stimm. Endliche Automaten und Zufallsfolgen. *Acta Informatica*, 1(4):345–359, 1972.

[50] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(Mar):371–421, 2008.

[51] Claude E Shannon. Mathematical theory of the differential analyzer. *Studies in Applied Mathematics*, 20(1-4):337–354, 1941.

[52] Alexander Shen, Vladimir A. Uspensky, and Nikolay Vereshchagin. *Kolmogorov Complexity and Algorithmic Randomness*, volume 220. American Mathematical Society, 2017.

[53] David Soloveichik. Personal communication, 2016.

[54] David Soloveichik, Matthew Cook, Erik Winfree, and Jehoshua Bruck. Computation with finite stochastic chemical reaction networks. *Natural Computing*, 7(4):615–633, 2008.

[55] David Soloveichik, Georg Seelig, and Erik Winfree. DNA as a universal substrate for chemical kinetics. *Proceedings of the National Academy of Sciences*, 107(12):5393–5398, 2010.

[56] Gerald Teschl. *Ordinary Differential Equations and Dynamical Systems*. American Mathematical Society, 2012.

[57] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(1):230–265, 1936.

[58] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. a correction. *Proceedings of the London Mathematical Society*, 43(2):544–546, 1937.

[59] Jean Ville. *Etude critique de la notion de collectif*. Gauthier-Villars Paris, 1939.

[60] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer Science & Business Media, 2005.

[61] Volodya Vovk, Alexander Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 444–453. Morgan Kaufmann Publishers Inc., 1999.

[62] Vladimir V. V'yugin. Ergodic theorems for individual random sequences. *Theoretical Computer Science*, 207(2):343–361, 1998.

[63] Yongge Wang. *Randomness and Complexity*. PhD thesis, University of Heidelberg, 1996.

[64] Klaus Weihrauch. *Computable Analysis: An Introduction*. Springer, 2000.

[65] Benjamin Weiss. *Single Orbit Dynamics*. Number 95 in Regional Conference Series in Mathematics. American Mathematical Society, 2000.

[66] Herrmann Weyl. *Philosophie der Mathematik und Naturwissenschaft: Nach der 2.* Walter de Gruyter GmbH & Co KG, 1927. Philosophy of Mathematics and Natural Science, Princeton University Press; with a new introduction by Frank Wilczek, 2009.

[67] Hisao Yamada. Real-time computation and recursive functions not real-time computable. *IRE Transactions on Electronic Computers*, EC-11(6):753–760, 1962.

[68] Alexander K. Zvonkin and Leonid A. Levin. The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms. *Russian Mathematical Surveys*, 25(6):83, 1970.